NBA game result prediction

Win rate estimation based on previous NBA players' and coaches' performance

Team members: Yichen Mu, Fangxu Zhou, Jingxuan Guo, Hao Fu

Catalog

- Introduction
- Datatset
- Goals & Initial ideas
  - Unsupervised Learning:
    - Detection outliers & Find outstanding players
  - Supervised Learning:
    - Prediction on outcomes of games
- Methods & Results
- Future work
- Difficulties encountered
- Summary

Introduction

- Basketball: a popular sport
- Machine Learning: a trendy science
- Combining together: finds amazing and fun

- Course project
- Explore what is learnt from CS542 & what is learnt from internet

Dataset

- Public
- From: http://www.cs.cmu.edu
- Only used for CS542 final project

Dataset

- Player regular season stats
- Player regular season career totals
- Player playoff career totals
- Team regular season stats
- NBA coaching records by season

Dataset: Features

- **Player regular season stats**
- **Player regular season career totals**
- **Player playoff career totals**
- Team regular season stats
- NBA coaching records by season

Contribution of each player during his each season game, total season career, total playoff carrer
(Pts: points, Dreb: defensive rebounds, etc.)

Dataset: Features

- **Player regular season stats**
- **Player regular season career totals**
- **Player playoff career totals**
- Team regular season stats
- NBA coaching records by season

PTS – Points
REB – Total Rebounds
ORB – Offensive Rebounds
DRB – Defensive Rebounds
AST – Assists
STL – Steals
BLK – Blocks
TO – Turnovers
PF – Personal Fouls
FGM – Total Field Goals Made
FGA – Total Field Goals Attempted
TPM– 3-Point Field Goals Made

ilkid	year	firstname	lastname	team	leag	gp	minutes	pts	oreb	dreb	reb	asts	stl	blk	turnover	pf	fga
ABRAMJO01	1946	John	Abramovic	PIT	N	47	0	527	0	0	0	35	0	0	0	161	834
AUBUCCH01	1946	Chet	Aubuchon	DE1	N	30	0	65	0	0	0	20	0	0	0	46	91
BAKERNO01	1946	Norm	Baker	CH1	N	4	0	0	0	0	0	0	0	0	0	0	1
BALTIHE01	1946	Herschel	Baltimore	ST1	N	58	0	138	0	0	0	16	0	0	0	98	263
BARRJO01	1946	John	Barr	ST1	N	58	0	295	0	0	0	54	0	0	0	164	438
BAUMHFR01	1946	Frankie	Baumholtz	CL1	N	45	0	631	0	0	0	54	0	0	0	93	856
BECKEMO01	1946	Moe	Becker	BOS	N	6	0	13	0	0	0	1	0	0	0	15	22

BOSTON UNIVERSITY

Dataset: Features

- Player regular season stats
- Player regular season career totals
- Player playoff career totals
- **Team regular season stats**
- NBA coaching records by season

Contributions of all the players of one team at one season game

Dataset: Features

team	year	leag	o_fgm	o_fga	o_ftm	o_fta	o_oreb	o_dreb	o_reb	o_asts	o_pf	o_stl	o_to	o_blk	o_3pm	o_3pa	o_pts	d_fgm	d_fga	d_ftm
BOS	1946	N	1397	5133	811	1375	0	0	0	470	1202	0	0	0	0	0	3605	0	0	0
CH1	1946	N	1879	6309	939	1550	0	0	0	436	1473	0	0	0	0	0	4697	0	0	0
CL1	1946	N	1674	5699	903	1428	0	0	0	494	1246	0	0	0	0	0	4251	0	0	0
DE1	1946	N	1437	5843	923	1494	0	0	0	482	1351	0	0	0	0	0	3797	0	0	0
NYK	1946	N	1465	5255	951	1438	0	0	0	457	1218	0	0	0	0	0	3881	0	0	0
PH1	1946	N	1510	5384	1098	1596	0	0	0	343	1082	0	0	0	0	0	4118	0	0	0
PIT	1946	N	1345	4961	984	1507	0	0	0	272	1360	0	0	0	0	0	3674	0	0	0
PRO	1946	N	1629	5582	1092	1666	0	0	0	481	1215	0	0	0	0	0	4350	0	0	0

- **Team regular season stats**
- NBA coaching records by season

Contributions of all the players of one team at one season game

Dataset: Features

- Player regular season stats
- Player regular season career totals
- Player playoff career totals
- Team regular season stats
- **NBA coaching records by season**

Counts of winning and lossing of a coach of a season game

Dataset: Features

coachid	year	yr_order	firstname	lastname	season_win	season_loss	playoff_win	playoff_loss	team
RUSSEJO01	1946	1	John	Russell	22	38	0	0	BOS
OLSENHA01	1946	1	Harold	Olsen	39	22	5	6	CH1
DEHNEDU01	1946	1	Dutch	Dehnert	17	20	0	0	CL1
CLIFFRO01	1946	2	Roy	Clifford	13	10	1	2	CL1
CURTIGL01	1946	1	Glenn	Curtis	12	22	0	0	DE1
SACHSPH01	1946	1	Philip	Sachs	8	18	0	0	DE1
COHALNE01	1946	1	Neil	Cohalan	33	27	2	3	NYK

- **NBA coaching records by season**

Counts of winning and lossing of a coach of a season game

Goals & Initial ideas

- Explore what we learnt as much as possible
- Divide mainly into two parts
  - Unsupervised learning
  - Supervised learning

Goals & Initial ideas

- Unsupervised learning
  - With clustering technique to find outliers among players (DBSCAN)
  - With clustering technique to find outstanding players (Kmeans++)
- Supervised learning

Goals & Initial ideas

- Unsupervised learning
- Supervised learning
  - Use data of players, coaches and teams of season games from previous years to predict the outcomes of season games of 2003
  - Using Kmeans++, KNN, SVM, Decision Tree

Methods & Results:

Unsupervised Learning: outliers detection

- Cluster the players using features of their contribution of their whole career to detect outliers.

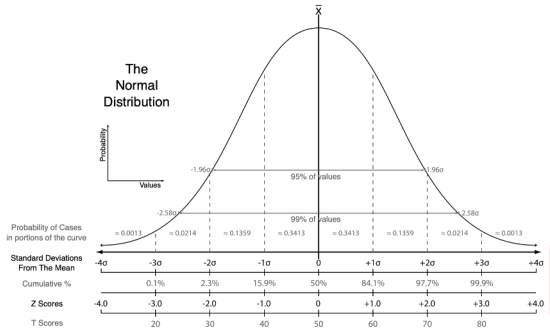
- Clean data
- Pre-handle data
- Reduce dimensions
- Cluster
- List outliers

Methods & Results:

Unsupervised Learning: outliers detection

- Clean data
  - Remove players with missing features
  - Remove unecessary features
  - List out the players who have 0 as value of the "minutes" feature
- Pre-handle data
  - Divide other features by "minutes"
  - Normalize: Standard-Score

$$z = \frac{x - \mu}{\sigma}$$

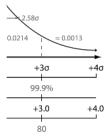


Metr

Un

- Cle
  -
  -
  -
- Pre
  -
  -

s" feature

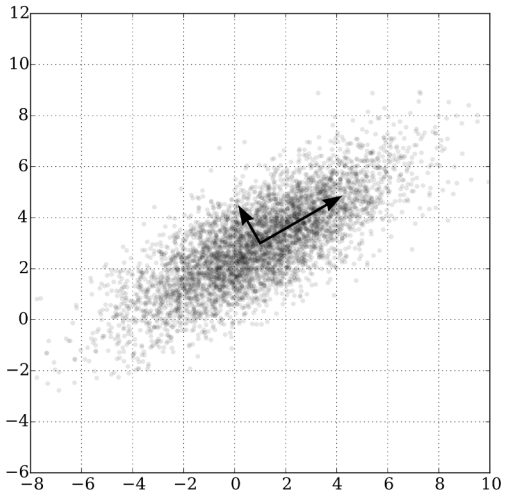


BOSTON UNIVERSITY

Methods & Results:

Unsupervised Learning: outliers detection

- Reduce dimensions: PCA
  - Project each data point onto only the first few principal components to obtain lower-dimensional data while preserving as much of the data's variation as possible

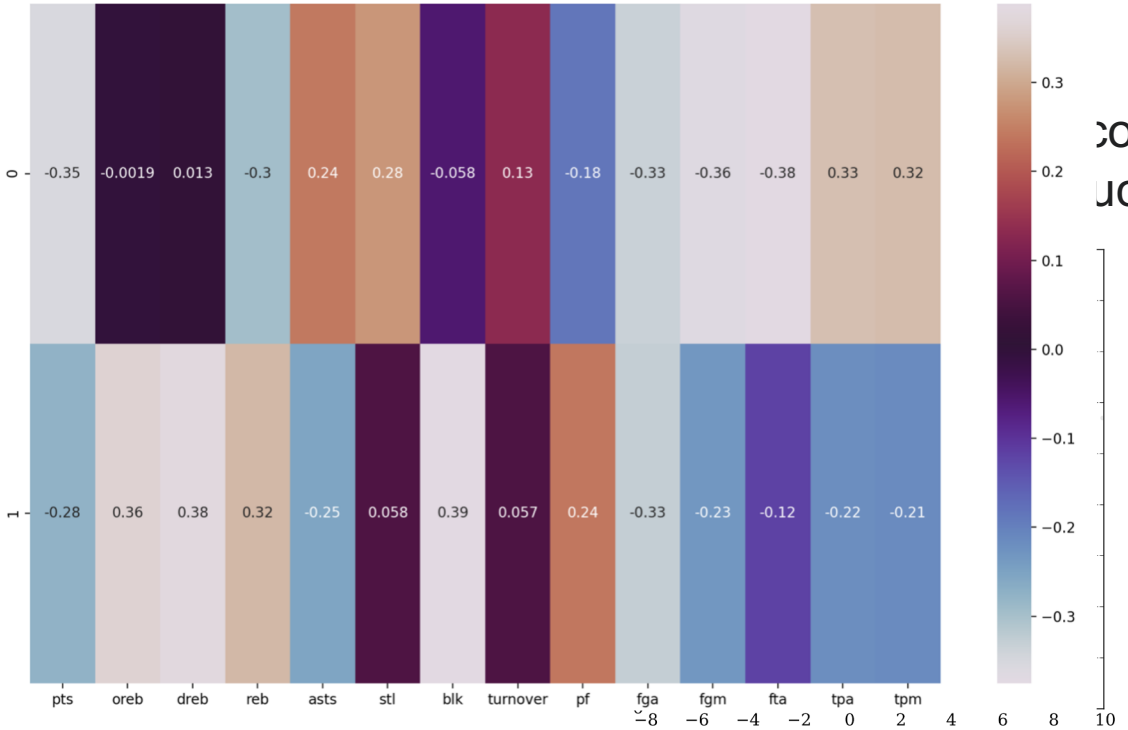


Methods & Results:

Unsupervised Learning: outliers detection

- Redu
  - Pro
    to c
    dat

components
uch of the

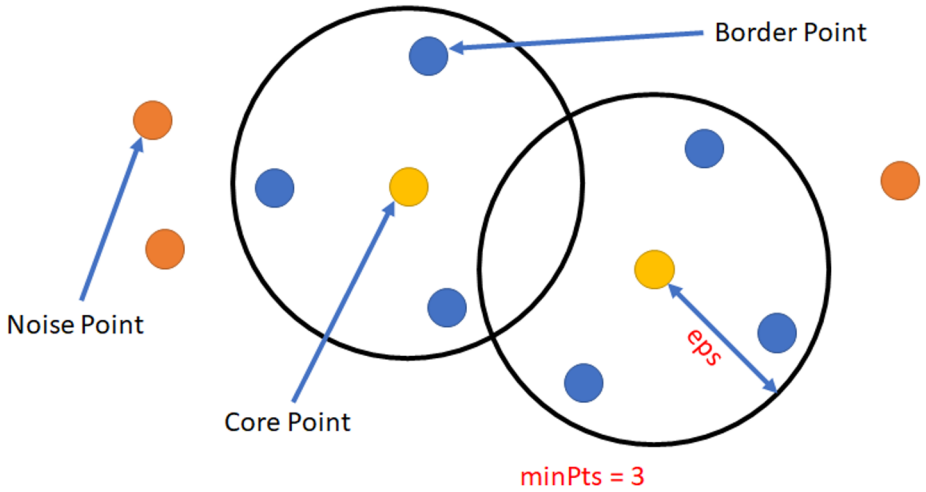


BOSTON UNIVERSITY

Methods & Results:

Unsupervised Learning: outliers detection

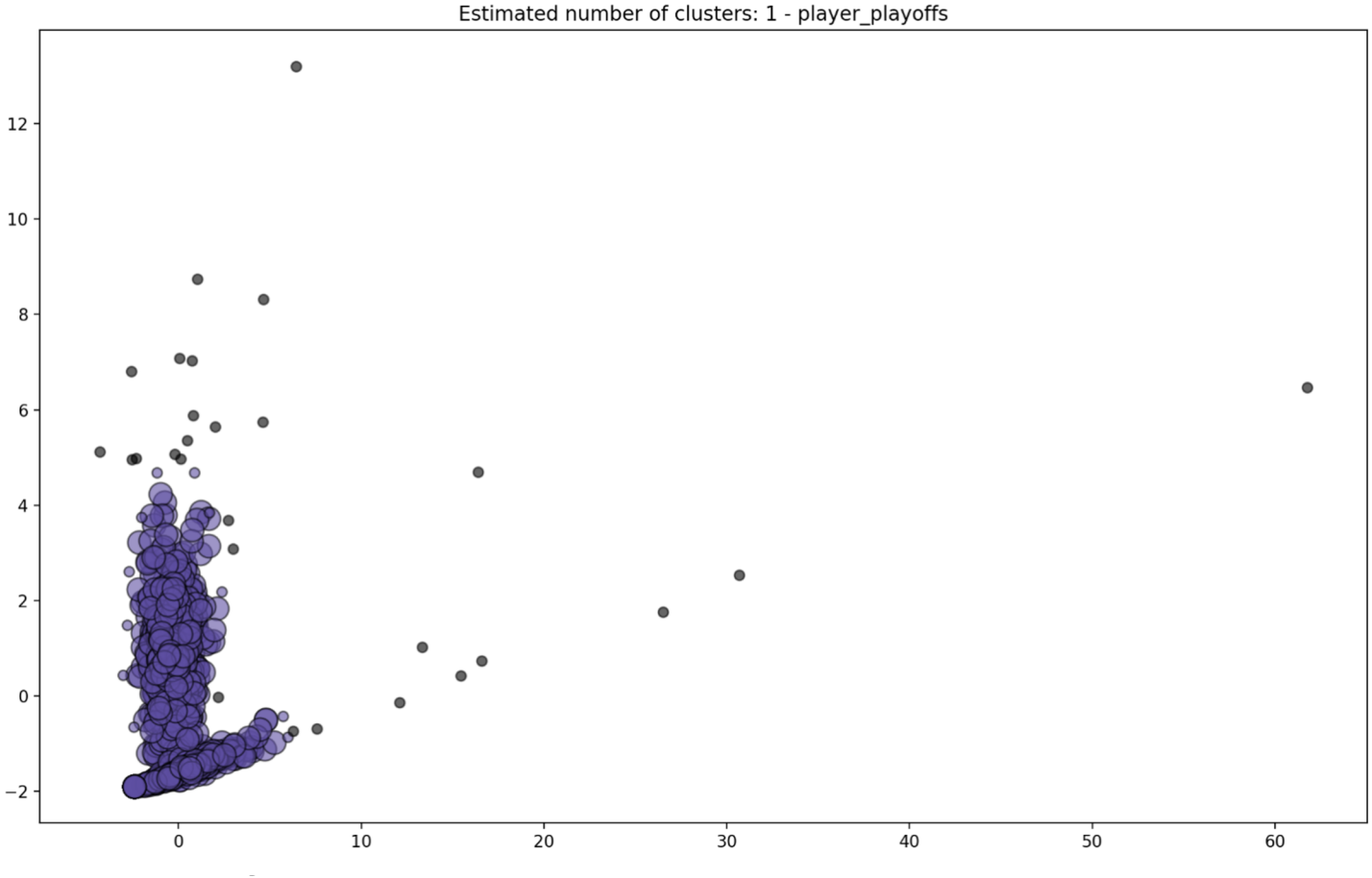
- Cluster: DBSCAN
  - Density-based spatial clustering of applications with noise
  - No need to pre-decide numbers of clusters
  - Choice of minPts and ε

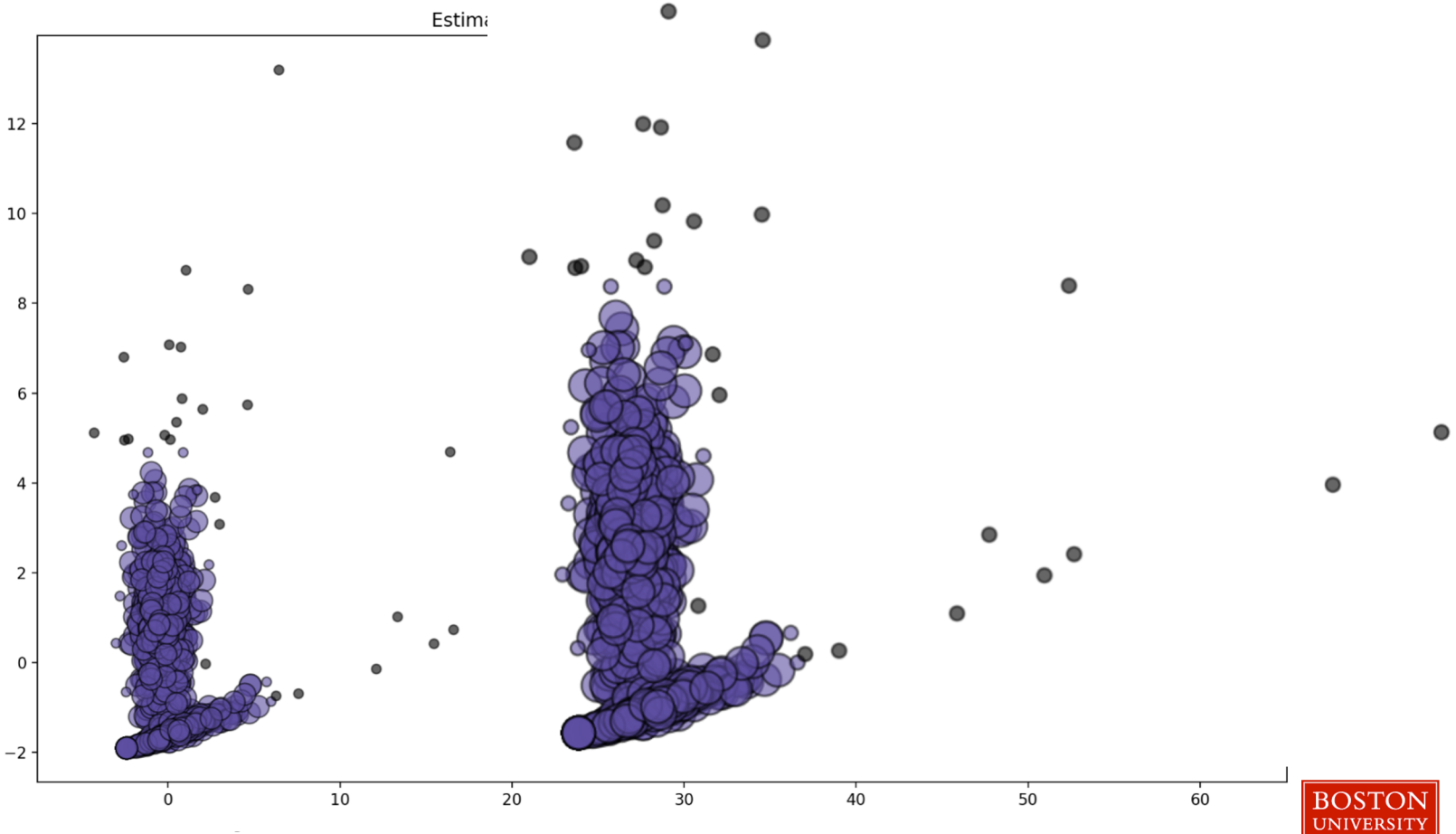


Methods & Results:

Unsupervised Learning: outliers detection

- Cluster: DBSCAN
  - Choice of minPts and ε
  - **Silhouette score**


Estimated number of clusters: 1 - player_playoffs
12
10
8
6
4
2
0
−2
0
10
20
30
40
50
60


Estima
12
10
8
6
4
2
0
−2
0
10
20
30
40
50
60

Methods & Results:

Unsupervised Learning: outliers detection

1	ilkid	firstname	lastname	gp	minutes	pts	dreb	oreb	reb	asts	stl	blk	turnover	pf	fga	fgm	fta	ftm	tpa	tpm
77	BREWEJA01	Jamison	Brewer	2	6	-0.68038635	3.54665183	6.44700639	3.88675404	0.83867720	-0.65438045	-0.36398182	3.20388489	2.69381024	-0.11531909	-1.18969088	1.44455845	1	-0.41238581	-0.2403
78	BRITTMI01	Mike	Brittain	1	2	1.76083703	-0.81729117	9.99981290	2.52086339	-0.82263293	-0.65438045	14.5980314	-0.77151527	-1.02339191	1.91887527	2.69096736	-0.81354132	0	-0.41238581	-0.2403
79	BROOKKE01	Kevin	Brooks	2	5	1.76083703	1.80107462	3.60476117	1.70132899	-0.82263293	-0.65438045	-0.36398182	-0.77151527	-1.02339191	3.13939189	1.91483571	1.89617841	1	2.83202264	-0.2403
80	CERVIAL01	Al	Cervi	27	116	4.28624054	-0.81729117	-0.65860664	1.46112063	8.28592951	-0.65438045	-0.36398182	-0.77151527	5.51375671	3.04995748	2.15570415	7.01194240	116	-0.41238581	-0.2403
81	CLOSSBI01	Bill	Closs	11	21	13.1997123	-0.81729117	-0.65860664	4.66726299	10.0945479	-0.65438045	-0.36398182	-0.77151527	10.3052241	16.7394342	12.1154231	11.4447146	31	-0.41238581	-0.2403
82	COOPEDU01	Duane	Cooper	2	4	-1.16863103	-0.81729117	9.99981290	2.52086339	1.66933226	-0.65438045	-0.36398182	-0.77151527	-1.02339191	3.44452105	-1.18969088	-0.81354132	0	11.7541459	-0.2403
83	EDMONKE01	Keith	Edmonson	1	2	1.76083703	-0.81729117	9.99981290	2.52086339	4.16129746	-0.65438045	-0.36398182	-0.77151527	-1.02339191	0.39322949	2.69096736	-0.81354132	0	-0.41238581	-0.2403
84	HAYESST01	Steve	Hayes	1	1	4.69030510	12.2745378	-0.65860664	6.61853535	-0.82263293	-0.65438045	-0.36398182	-0.77151527	-1.02339191	1.91887527	6.57162561	-0.81354132	0	-0.41238581	-0.2403
85	HEALSH01	Shane	Heal	2	3	4.69030510	-0.81729117	-0.65860664	-1.57680857	2.49998733	-0.65438045	-0.36398182	-0.77151527	-1.02339191	0.90177808	3.98452011	-0.81354132	0	10.4023090	19.7999
86	HENRYCA01	Carl	Henry	1	2	3.22557106	-0.81729117	-0.65860664	-1.57680857	-0.82263293	-0.65438045	-0.36398182	-0.77151527	-1.02339191	0.39322949	2.69096736	-0.81354132	0	7.69863533	14.7898
87	HOLLAJO01	Joe	Holland	21	37	6.66967542	0.24420847	0.49365492	2.18861971	7.25941635	4.25627905	-0.36398182	1.80712267	8.42031086	7.77405636	8.03998278	3.39750962	11	-0.41238581	-0.2403
88	HOLZMRE01	Red	Holzman	24	79	2.61372014	-0.81729117	-0.65860664	1.12039322	2.07940248	-0.65438045	-0.36398182	-0.77151527	1.23516129	2.92309780	2.54360059	3.13098740	26	-0.41238581	-0.2403
89	JOHNSAR01	Arnie	Johnson	22	166	2.64320693	-0.81729117	-0.65860664	6.17420948	3.32063450	-0.65438045	-0.36398182	-0.77151527	3.72387831	1.55124978	1.70911407	4.16516060	92	-0.41238581	-0.2403
90	JONESWA02	Wallace	Jones	6	8	26.2951321	-0.81729117	-0.65860664	-1.57680857	26.5889842	-0.65438045	-0.36398182	-0.77151527	24.9970232	27.8548534	21.1240940	27.9772309	29	-0.41238581	-0.2403
91	JUDKIJE01	Jeff	Judkins	7	10	1.46789022	0.49189172	5.73644508	1.70132899	-0.82263293	2.97950758	-0.36398182	-0.77151527	-1.02339191	1.30861696	1.91483571	-0.81354132	0	4.45422687	2.76571
92	LAUDEPR01	Priest	Lauderdale	3	7	-1.16863103	1.05297011	2.38665608	0.76471826	-0.82263293	-0.65438045	-0.36398182	9.45094231	0.03866584	0.17528009	-1.18969088	-0.81354132	0	-0.41238581	-0.2403
93	MCKINHO01	Horace	Mckinney	7	20	5.56914552	-0.81729117	-0.65860664	4.97946657	5.65647658	-0.65438045	-0.36398182	-0.77151527	9.75649434	7.41120007	6.18355978	2.57360834	8	-0.41238581	-0.2403
94	MEYERLO01	Loren	Meyer	3	14	-1.16863103	2.92323140	2.38665608	1.93548168	-0.11064287	-0.65438045	3.91087912	4.33971351	-1.02339191	-0.26061869	-1.18969088	-0.81354132	0	0.74633149	-0.2403
95	MOSLEGL01	Glenn	Mosley	3	6	1.27259235	-0.81729117	-0.65860664	-0.21091792	0.83867720	-0.65438045	4.62335594	-0.77151527	-1.02339191	0.39322949	1.39741461	2.57360834	1	-0.41238581	-0.2403
96	MURRAKE01	Ken	Murray	6	15	6.64328381	-0.81729117	-0.65860664	6.07217909	5.82260759	-0.65438045	-0.36398182	-0.77151527	5.41975849	10.6659110	7.60646781	2.34779836	6	-0.41238581	-0.2403
97	RANDAMA01	Mark	Randall	2	6	-1.16863103	7.91059484	2.89419987	5.25264470	-0.82263293	-0.65438045	4.62335594	3.20388489	0.21567547	-0.62386768	-1.18969088	-0.81354132	0	-0.41238581	-0.2403
98	RATKOGE01	George	Ratkovicz	24	59	11.6912372	-0.81729117	-0.65860664	10.2300428	5.59734520	-0.65438045	-0.36398182	-0.77151527	10.8212522	8.95236088	9.33412809	16.1796163	99	-0.41238581	-0.2403
99	RAUTILE01	Leo	Rautins	3	5	0.58904980	-0.81729117	7.86812899	1.70132899	1.17093922	6.61339561	-0.36398182	-0.77151527	1.95036981	0.69835865	0.36257241	-0.81354132	0	6.07643110	5.77175
100	SCHEFTO01	Tom	Scheffler	3	10	0.88199661	1.80107462	5.73644508	2.52086339	-0.82263293	2.97950758	-0.36398182	1.61372482	-1.02339191	-0.21702881	0.36257241	1.89617841	3	-0.41238581	-0.2403
101	STROEJO01	John	Stroeder	1	1	7.61977317	-0.81729117	-0.65860664	-1.57680857	-0.82263293	-0.65438045	-0.36398182	-0.77151527	-1.02339191	1.91887527	6.57162561	-0.81354132	0	15.8096564	29.8200
102	VAUGHDA02	David	Vaughn	1	1	-1.16863103	-0.81729117	-0.65860664	-1.57680857	-0.82263293	-0.65438045	29.5600447	-0.77151527	-1.02339191	-1.13241628	-1.18969088	-0.81354132	0	-0.41238581	-0.2403
103	WARDHE01	Henry	Ward	1	1	10.5492412	-0.81729117	-0.65860664	-1.57680857	-0.82263293	-0.65438045	-0.36398182	-0.77151527	-1.02339191	8.02145838	14.3329421	-0.81354132	0	-0.41238581	-0.2403
104	WHEATDE01	Dejuan	Wheat	1	3	0.78434767	3.54665183	-0.65860664	1.15497273	-0.82263293	11.4585796	-0.36398182	-0.77151527	-1.02339191	0.90177808	1.39741461	-0.81354132	0	-0.41238581	-0.2403

Methods & Results:

Unsupervised Learning: outliers detection

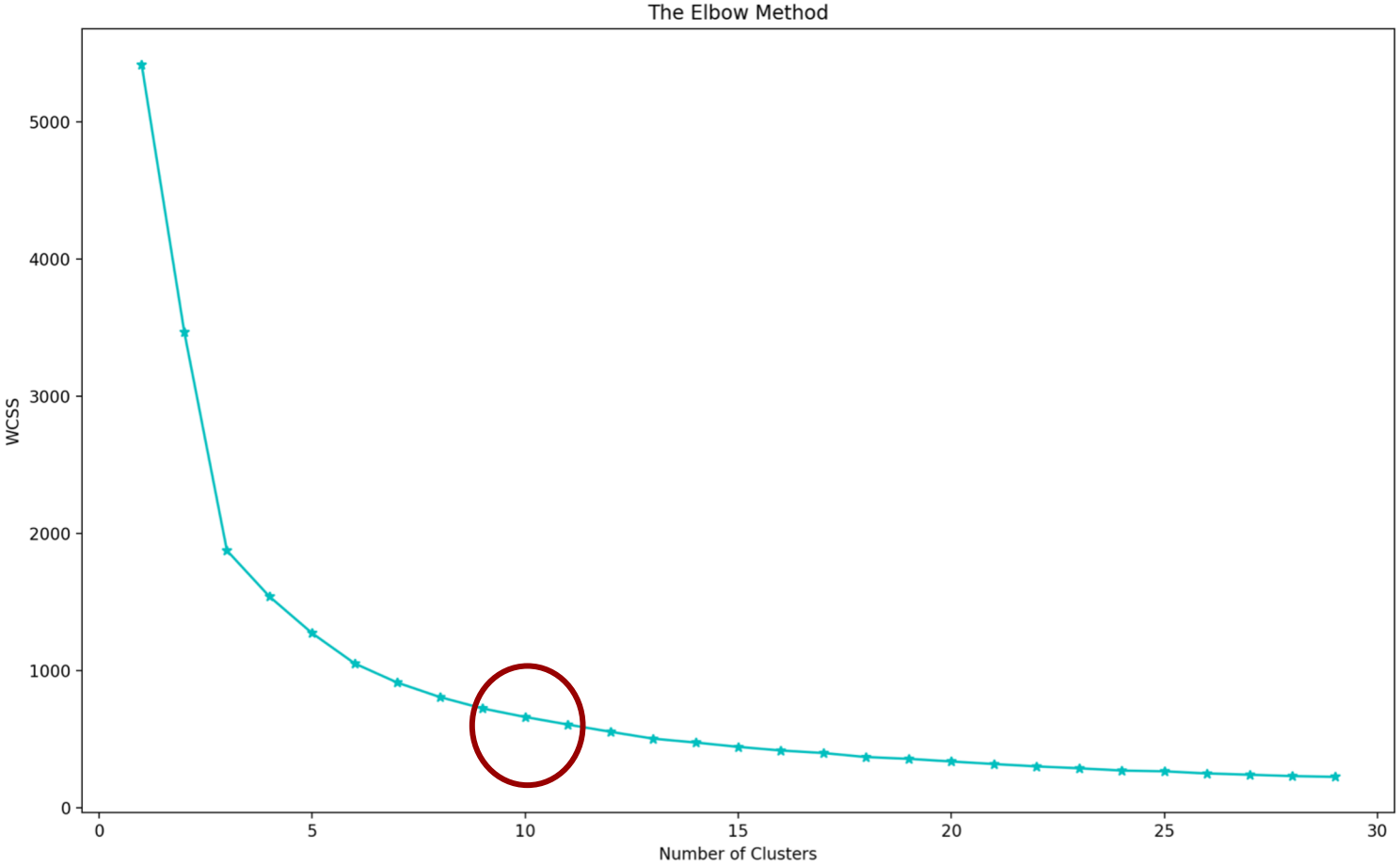
1	ilkid	firstna
77	BREWEJA01	Jamiso
78	BRITTMI01	Mike
79	BROOKKE01	Kevin
80	CERVIAL01	Al
81	CLOSSBI01	Bill
82	COOPEDU01	Duane
83	EDMONKE01	Keith
84	HAYESST01	Steve
85	HEALSH01	Shane
86	HENRYCA01	Carl
87	HOLLAJO01	Joe
88	HOLZMRE01	Red
89	JOHNSAR01	Arnie
90	JONESWA02	Wallac
91	JUDKIJE01	Jeff
92	LAUDEPR01	Priest
93	MCKINHO01	Horace
94	MEYERLO01	Loren
95	MOSLEGL01	Glenn
96	MURRAKE01	Ken
97	RANDAMA01	Mark
98	RATKOGE01	George
99	RAUTILE01	Leo
100	SCHEFTO01	Tom
101	STROEJO01	John
102	VAUGHDA02	David
103	WARDHE01	Henry
104	WHEATDE01	Dejuan

90	JONESWA02	Wallace	Jones	6	8	26.2951321	ftm	tpa	tpm
91	JUDKIJE01	Jeff	Judkins	7	10	1.46789022	1	-0.41238581	-0.2403
92	LAUDEPR01	Priest	Lauderdale	3	7	-1.1686310	0	-0.41238581	-0.2403
93	MCKINHO01	Horace	Mckinney	7	20	5.56914552	1	2.832022646	-0.2403
94	MEYERLO01	Loren	Meyer	3	14	-1.1686310	116	-0.41238581	-0.2403
95	MOSLEGL01	Glenn	Mosley	3	6	1.27259235	31	-0.41238581	-0.2403
96	MURRAKE01	Ken	Murray	6	15	6.64328381	0	11.75414591	-0.2403
97	RANDAMA01	Mark	Randall	2	6	-1.1686310	0	-0.41238581	-0.2403
98	RATKOGE01	George	Ratkovicz	24	59	11.6912372	0	-0.41238581	-0.2403
99	RAUTILE01	Leo	Rautins	3	5	0.58904980	0	10.40230905	19.7999
100	SCHEFTO01	Tom	Scheffler	3	10	0.88199661	0	7.698635338	14.7898
101	STROEJO01	John	Stroeder	1	1	7.61977317	11	-0.41238581	-0.2403
							26	-0.41238581	-0.2403
							92	-0.41238581	-0.2403
							29	-0.41238581	-0.2403
							0	4.454226877	2.76571
							0	-0.41238581	-0.2403
							8	-0.41238581	-0.2403
							0	0.746331497	-0.2403
							1	-0.41238581	-0.2403
							6	-0.41238581	-0.2403
							0	-0.41238581	-0.2403
							99	-0.41238581	-0.2403
							0	6.076431108	5.77175
							3	-0.41238581	-0.2403
							0	15.80965649	29.8200
							0	-0.41238581	-0.2403
							0	-0.41238581	-0.2403
							0	-0.41238581	-0.2403

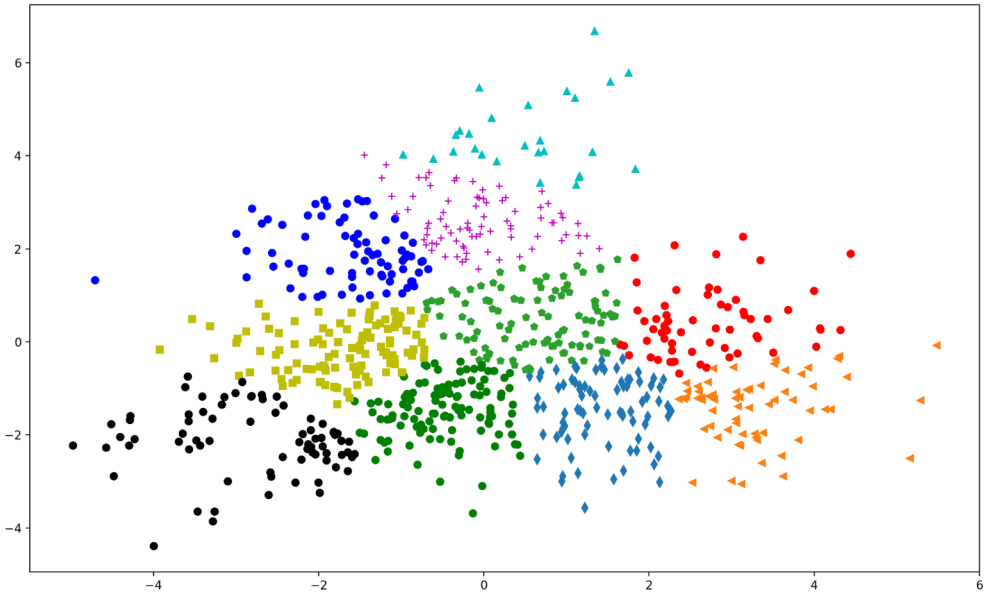
Methods & Results:

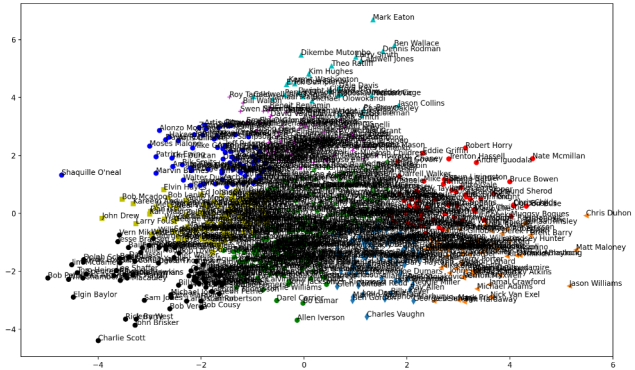
Unsupervised Learning: to find outstanding players

- Data pre-processing: same as "outliers detection"
- Elbow method
- Kmeans++


The Elbow Method
WCSS
Number of Clusters
0
1000
2000
3000
4000
5000
0
5
10
15
20
25
30

Methods & Results:





BOSTON UNIVERSITY

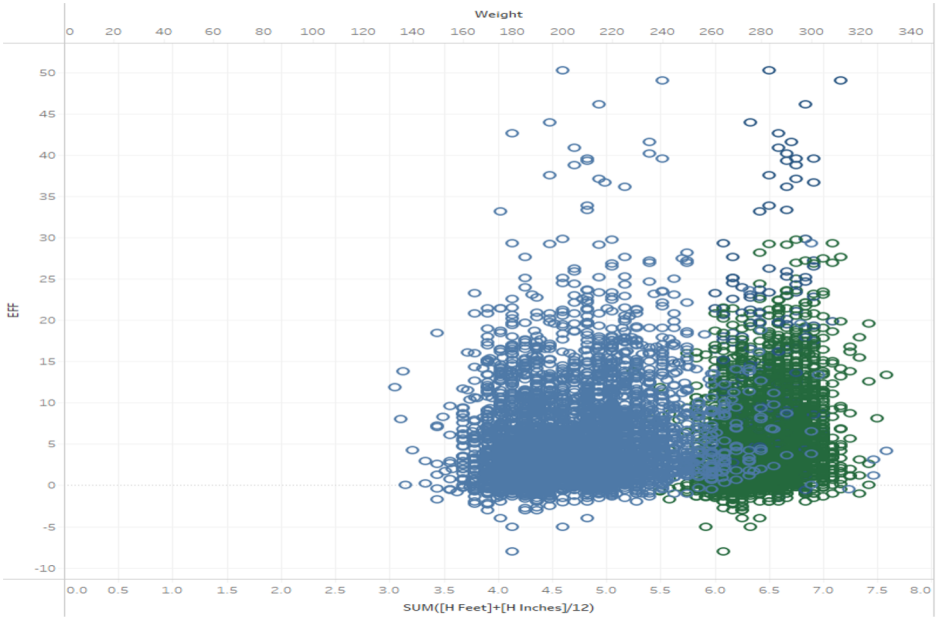
Then, How could we find out the winner?

Preliminary ideas

- Many gambling sites have their own methods of forecasting
  - More recent data
  - Some inside infomation
  - Use crawlers to crawl Facebook feeds to assess players' mental states

- Our ideas (under the limitation of the dataset)
  - physical index
  - Athlete service time
  - Athlete Career time

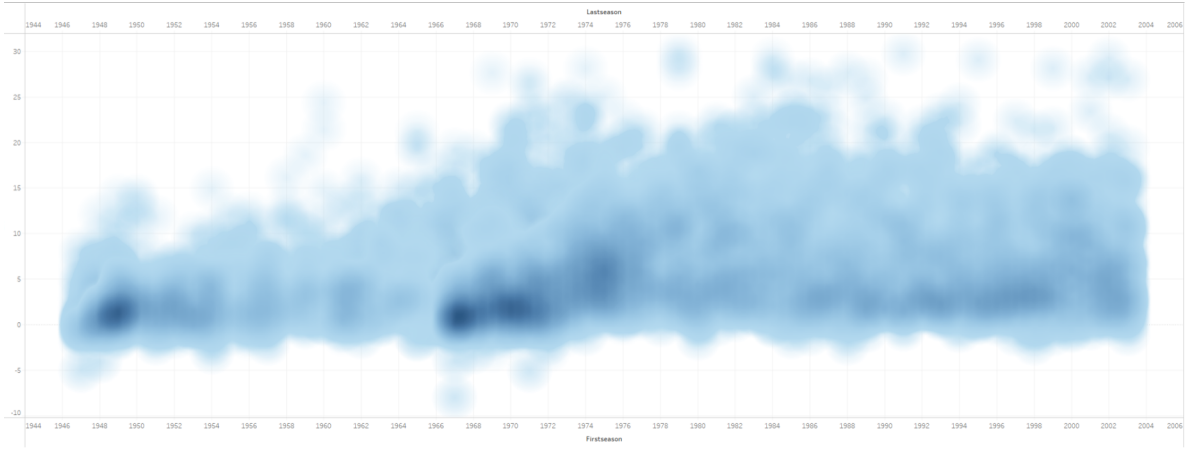
Idea test-physical index



- Blue is their weight.(170-270)pound
- Green is their height.(5.6-7.1)feed
- Could not find out apperant relation between performance and physical index

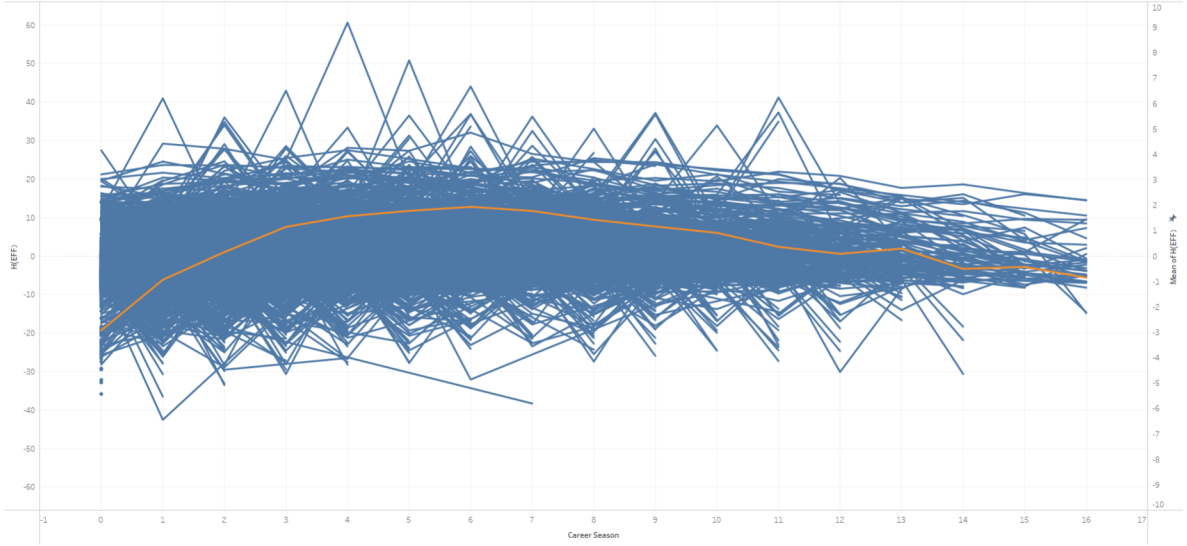
BOSTON UNIVERSITY

Idea test-Service time



- Could not find out apparent relation between performance and service time

Idea tests-Career time



- Each blue lines show the data of the player's performance minus the average performance of those years.
- The orange line shows the expectation of the performance based on Career time.
- Their career time may affect their performance.

Methods & Results:

Supervised Learning: prediction of outcomes

Then now in order to know more concrete winning rates for every team, what should we do?

Overall idea:

We preprocess the data, then use the machine learning algorithms to learn and fit some different models to predict the winning percentage for every team.

Methods & Results: prediction of outcomes

Working process

Data pre-processing

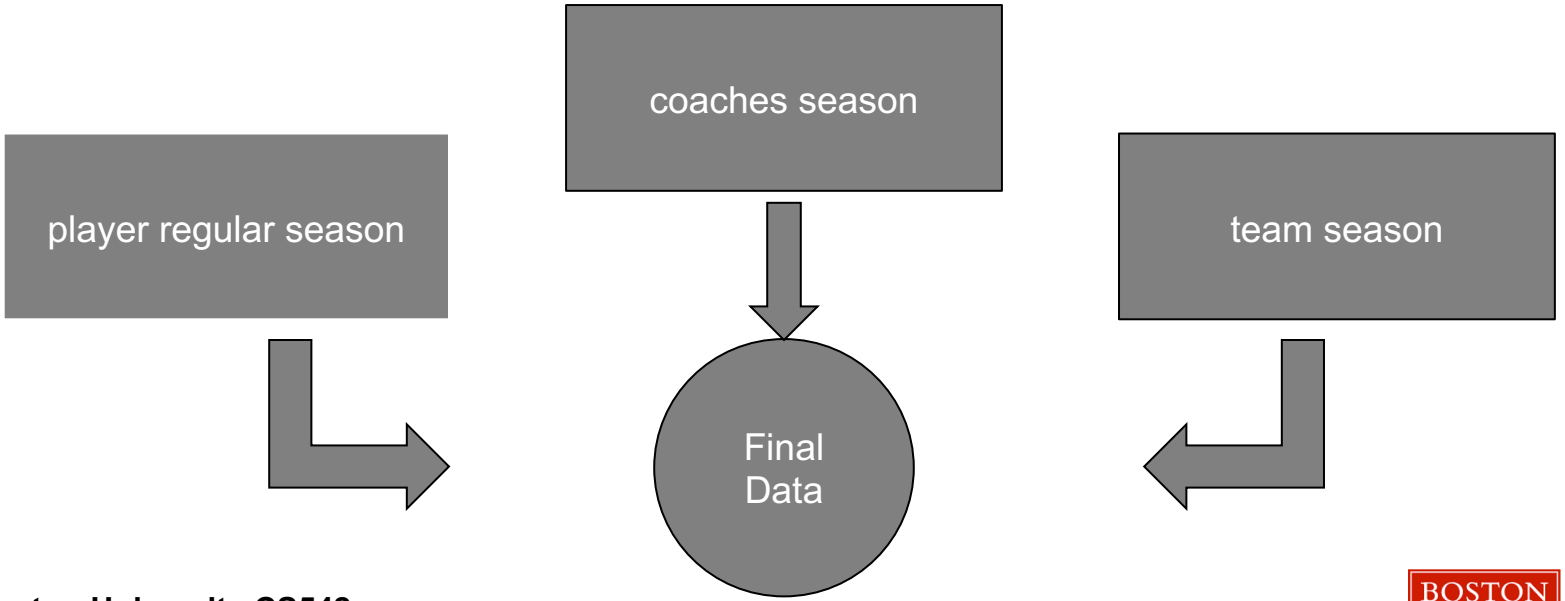
K-means++ clustering algorithm

Supervised learning

Result Comparing

Methods & Results: prediction of outcomes

Data pre-processing: Collect and merge



Methods & Results: prediction of outcomes

Data pre-processing: Filter

Divide the original dataset into two parts

1. Training data (data before 2003) for later learning.
2. Testing data (data in 2003) for test.

Methods & Results: prediction of outcomes

Data pre-processing: Calculate

Create some special representative features

1. Efficiency of Players and teams: ([Pts]+[Asts]+[Dreb]+[Oreb]+[Stl]+[Blk]
   ([Fga]-[Fgm])-([Fta]-[Ftm])-[Turnover])/[Gp]
2. Efficiency of coaches:
   ([Season_win]*2+[Season_loss]*0)/([Season_win]+[Season_loss])
3. Rate of Fg: ([Fgm]/[Fga])
4. Rate of Ft: ([Ftm]/[Fta])
5. Efficiency of pts: ([pts]/[minutes])

   ...

Methods & Results: prediction of outcomes

Data pre-processing: Normalize

After dropping some non-numeric columns, we use the preprocessing.MinMaxScaler()

Methods & Results: prediction of outcomes

K-means++ clustering algorithm

Why do we have to use K-means++ clustering algorithm?

player_regular_season

ilkid	year	firstname	lastname	team	leag	gp	minutes	pts	oreb	dreb	reb	asts	stl	blk	turnover	pf	fga	fgm	fta	ftm	tpa	tpm
ABRAMJO01	1946	John	Abramovic	PIT	N	47	0	527	0	0	0	35	0	0	0	161	834	202	178	123	0	0
AUBUCCH01	1946	Chet	Aubuchon	DE1	N	30	0	65	0	0	0	20	0	0	0	46	91	23	35	19	0	0
BAKERNO01	1946	Norm	Baker	CH1	N	4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
BALTIHE01	1946	Herschel	Baltimore	ST1	N	58	0	138	0	0	0	16	0	0	0	98	263	53	69	32	0	0
BARRJO01	1946	John	Barr	ST1	N	58	0	295	0	0	0	54	0	0	0	164	438	124	79	47	0	0
BAUMHFR01	1946	Frankie	Baumholtz	CL1	N	45	0	631	0	0	0	54	0	0	0	93	856	255	156	121	0	0

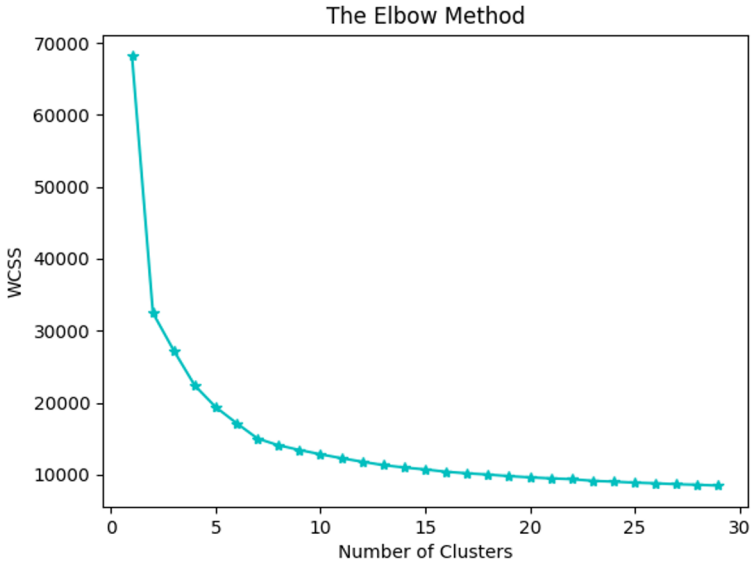
Methods & Results: prediction of outcomes

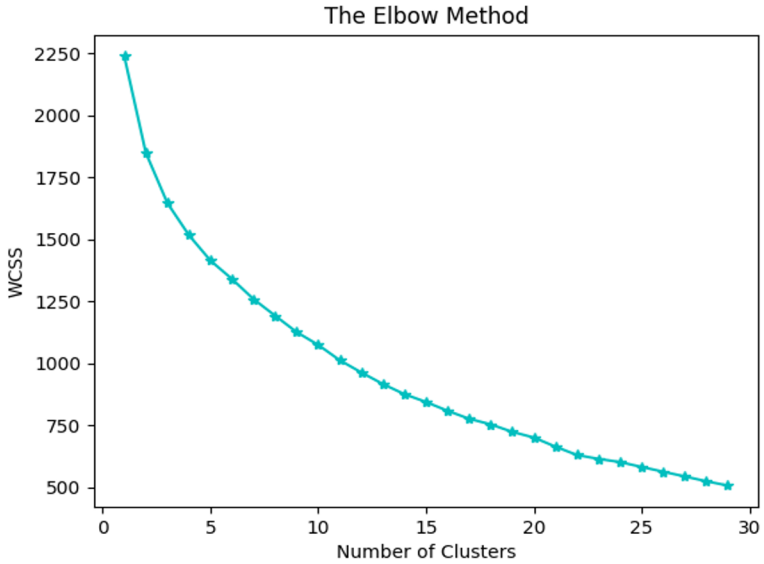
K-means++ clustering algorithm

- Data normalization
  - preprocessing.MinMaxScaler()
- Related material
  - Elbow function (self-defined)
  - Create_labels function (self-defined)
  - Kmeans++ algorithm
  - matplotlib package

Methods & Results: prediction of outcomes

K-means++ clustering algorithm





Methods & Results: prediction of outcomes

Added labels Datasets (X_train, Y_train, X_test, Y_test)

```
      year  o_fgm  o_fga  o_ftm  o_fta  o_oreb  o_dreb  o_reb  o_asts  o_pf  o_stl  o_to  o_blk  o_3pm  o_3pa  o_pts  d_fgm  d_fga
   d_ftm  d_fta  d_oreb  d_dreb  d_reb  d_asts  d_pf  d_stl  d_to  d_blk  d_3pm  d_3pa  d_pts      pace  won  lost  yr_order
  season_win  season_loss  gp  minutes  pts  oreb  dreb  reb  asts  stl  blk  turnover  pf  fga  fgm  fta  ftm  tpa  tpm
  T_o_Effic  T_o_FG_rate  T_o_FT_rate  T_o_3P_rate  T_d_Effic  T_d_FG_rate  T_d_FT_rate  T_d_3P_rate  T_WIN_LOSE_rate  Coach_Effic
   P_Season_Effic  P_Season_FG_rate  P_Season_FT_rate  P_Season_PTS_rate
0      1946   1397   5133    811   1375       0       0      0     470  1202     0     0     0     0     0   3605     0     0
       0      0       0       0      0       0     0      0     0      0      0     0  3900  0.000000   22    38        1
      22          38   6       0    13      0     0     0     1     0     0       0.0   15    22    5    4    3    0    0       -225
      0.272161     0.589818     0.000000       3900     0.000000     0.000000     0.000000         0.366667     0.733333
  -0.666667          0.227273          0.750000          0.000000
1      1946   1397   5133    811   1375       0       0      0     470  1202     0     0     0     0     0   3605     0     0
       0      0       0       0      0       0     0      0     0      0      0     0  3900  0.000000   22    38        1
      22          38  58       0   567      0     0     0    60     0     0       0.0  115   870   223  193  121    0    0       -225
      0.272161     0.589818     0.000000       3900     0.000000     0.000000     0.000000         0.366667     0.733333
  -1.586207          0.256322          0.626943          0.000000
```

```
0        4
1        4
2        4
3        4
4        4
        ..
18105    3
18106    4
18107    1
18108    4
18109    4
Name: label
```


BOSTON UNIVERSITY

Methods & Results: prediction of outcomes

Supervised learning

- Model construction: Makepipeline()
  - MinMaxScaler()
  - PCA
  - Learning algorithms
- Learning algorithm
  - KNN
  - SVM
  - Decision Tree
  - Neural Network

Methods & Results: prediction of outcomes

Selections about parameters

PCA: n_components=0.98, svd_solver='auto'
KNN: n_neighbors=9
Decision Tree:
criterion='entropy',max_depth=None,min_samples_split=2,min_samples_leaf=1,max_features=None,max_leaf_nodes=None,min_impurity_decrease=0
Support Vector Machine: kernel='rbf',class_weight='balanced',C=3.0
Neural Network: solver='sgd',activation='relu',alpha=1e-4,hidden_layer_sizes=(300,300,300),random_state=1,max_iter=200,learning_rate_init=0.001

Methods & Results: prediction of outcomes

Accuracy Results and comparison

When PCA (n_components=0.98, svd_solver='auto'),
KNN: RMSE on testing set = 1.3280510201777347
Decision Tree: RMSE on testing set = 1.3043080587787235
SVM: RMSE on testing set = 1.86429873038029
Neural Network: RMSE on testing set = 1.115986930599231

Future Work

- Finish the result-analysis phase of the unsupervised learning part
- Collecting more data to make our dataset more complete.

Diffculties Encountered

- Incomplete data
- Outdated data
- Unfamiliar to Pandas, Numpy and some other python packages

Summary

Thank you!

Resources

- https://towardsdatascience.com/which-nba-players-are-most-similar-machine-learning-provides-the-answers-r-project-b903f9b2fe1f
- https://ruder.io/semi-supervised/
- https://donernesto.github.io/blog/outlier-detection-with-dbscan/
- https://per48.co/blog/outliers_nba/
- https://nikkimarinsek.com/blog/7-ways-to-label-a-cluster-plot-python


Q&A