NBA Salary Prediction

Brandon Arcilla

Problem

Can I develop a classification method for NBA player salary?

Why?

NBA Players want to find out how much they would be worth to a team and NBA teams can find out which players are outperforming or underperforming their peers that are in the same salary range.

How?

Using player data (age, position, and salary) and NBA advanced statistics along with various classification models.

Data Wrangling

Where'd you get the data?

- Scraped: www.basketball-reference.com (Player statistics)
- Downloaded: data.world (Salary)

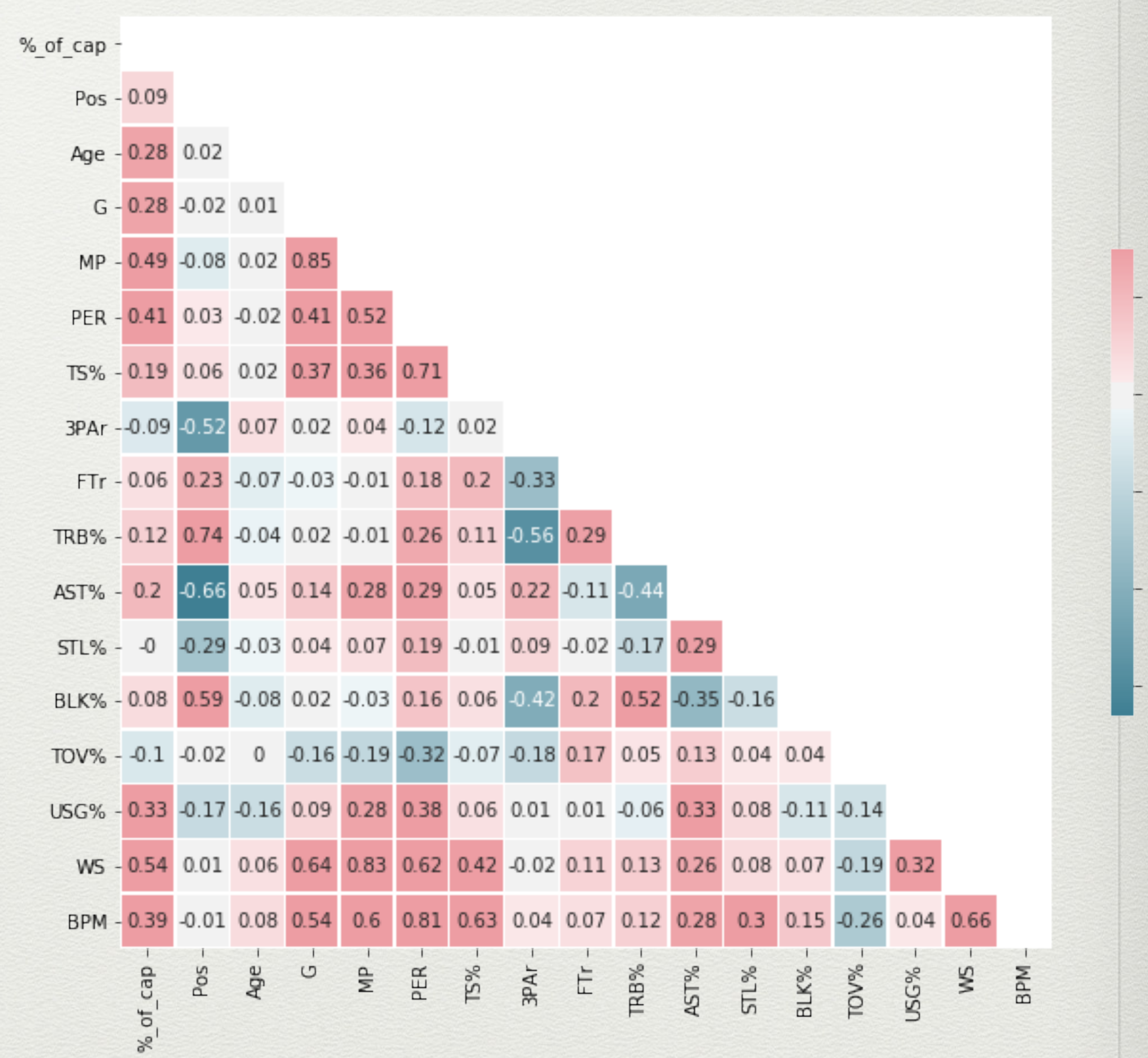
What did you do with data?

- Removed blank columns and replaced NaN values with 0.
- Computed player salary as a percent of the NBA teams salary cap

Exploratory Data Analysis

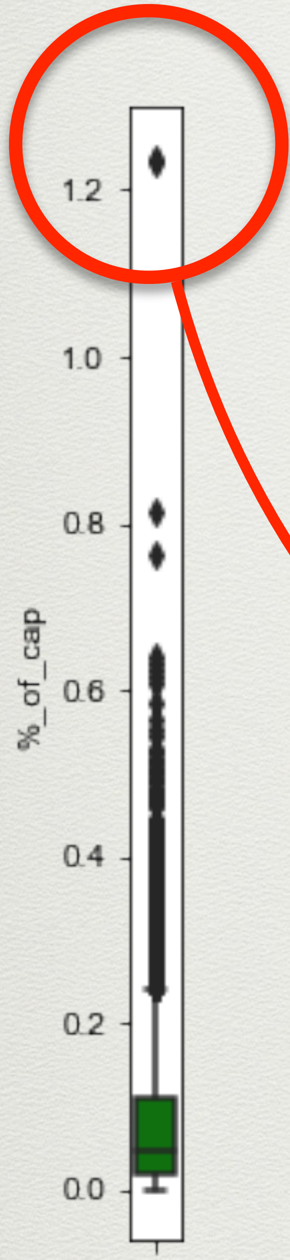
What did the data show you?

Moderate Positive Correlation (0.3 to 0.7)	Minutes Played, Player Efficiency Rating, Usage %, Win Shares, Box Plus/Minus
Weak Positive Correlation (0 < c ≤ 0.3)	Position, Age, Games Played, True Shooting %, Free Throw Rate, Total Rebounding %, Assist %, Block %
Weak Negative Correlation (-0.3 ≤ c < 0)	3-Point Attempt Rate, Turnover %
No Correlation (0)	Steal %

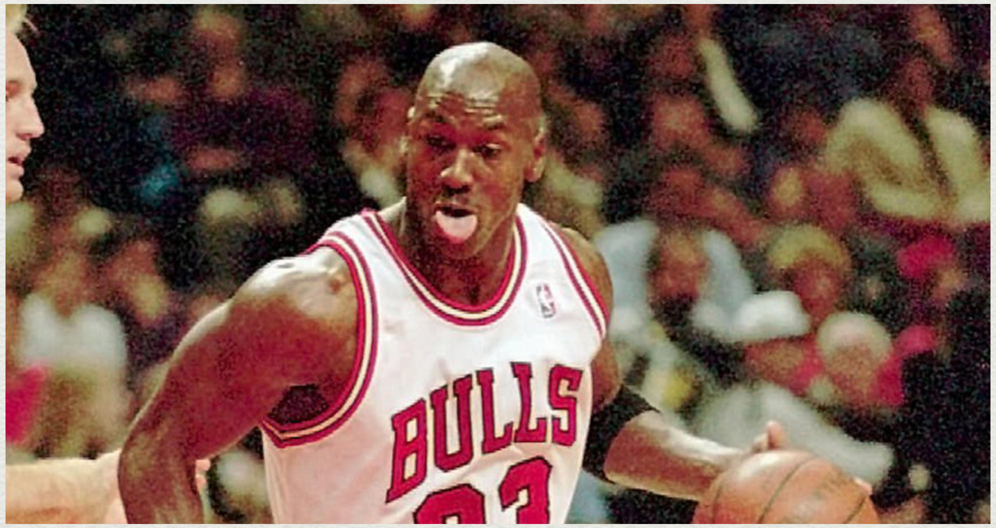


Exploratory Data Analysis

What did else the data show you?



- Outliers can be common in basketball statistics and will not be disregarded.
- There's been only 1 player who has had a yearly salary that was greater than the team salary.



In-depth Analysis

What do we do first?

- Build the dataset

Preprocess Steps	On what?	Why?	How?
One-Hot Encoding	Position	Make numerical and increase dimensions	pd.get_dummies
Binning	Salary Cap %	Convert ranges into categorical features	pd.cut
MinMaxScaler	All Data	Increase speed of learning	MinMaxScaler

	Age	G	MP	PER	TS%	3PAr	FTr	TRB%	AST%	STL%	...	TOV%	USG%	WS	BPM	C	PF	PG	SF	SG	group
0	26	57	2029	18.6	0.535	0.324	0.164	3.9	34.7	1.6	...	10.1	25.1	5.5	1.1	0	0	1	0	0	10%-15%
1	31	53	516	9.6	0.489	0.055	0.133	10.6	9.2	0.9	...	13.4	17.7	0.2	-5.2	0	0	0	1	0	0-10%
2	22	60	560	8.7	0.506	0.426	0.161	4.2	31.6	2.5	...	29.1	18.8	0.2	-4.7	0	0	1	0	0	0-10%
3	23	41	362	7.8	0.447	0.314	0.343	4.1	20.8	3.0	...	22.0	19.0	-0.3	-6.1	0	0	0	0	1	0-10%
4	23	73	1614	11.8	0.518	0.008	0.289	10.8	6.5	1.8	...	11.9	13.7	1.7	-1.8	1	0	0	0	0	0-10%

Sample output (MinMaxScaler not applied here)

In-depth Analysis (cont.)

What models will be used?

- 6 Classification Models will be evaluated

1. Decision Tree Classifier
2. Random Forest Classifier
3. Support Vector Machine Classifier
4. AdaBoost Classifier
5. XGBoost Classifier

In-depth Analysis (cont.)

What's next?

- Create Features and Labels from dataset
- Split Features and Label to train and test datasets
- Use KFold cross validation on train dataset to evaluate each models performance in respect to average fit time, average train score, average test score, and test score standard deviation

	classifier	mean_fit_time	mean_test_score	std_test_score	mean_train_score
0	DecisionTree	0.095745	0.666398	0.007840	1.000000
1	RandomForest	0.174659	0.745540	0.009935	0.983501
2	SVC	1.227508	0.730651	0.011307	0.731388
3	AdaBoost	0.874373	0.747150	0.009784	0.758417
4	XGBoost	5.135772	0.758417	0.011314	0.793427

Model	Fit Time Rank	Test Score Rank	Std Rank	Analysis	Dive Deeper?	Why?
Decision Tree	1	5	1	Lowest test score, low test variance, short fit time, overfits on train	No	All models test better and do not overfit as
Random Forest	2	3	3	High test score, low test variance, short fit time, overfits on train	Yes	High testing accuracy, low fit time
SVC	4	4	4	High test score, long fit time	No	Hard to extract feature importance, fit time is
AdaBoost	3	2	2	High test score, low variance, ok fit time, does not overfit as much	Yes	High test score, low variance, does not
XGBoost	5	1	5	High test score, long fit time, overfits on train	No	Long fit time

In-depth Analysis (cont.)

How can we improve?

- Fit Random Forest and AdaBoost with train data and then score on test data with default hyperparameter values to obtain accuracy score baseline.

	classifier	accuracy_score
0	RandomForest	0.757344
1	AdaBoost	0.755332

- We want to tune the hyperparameters for each model to improve the accuracy score
- GridSearchCV will be used to find optimal hyperparameters

In-depth Analysis (cont.)

Model	Best Parameters	Tuned Accuracy Score	Default Accuracy Score
Random Forest	max_depth = 0.1 max_features = auto n_estimator = 1000	0.763	0.757
AdaBoost	learning_rate = 0.1 n_estimator = 1000	0.764	0.755

What happened?

Both models improved with the given hyperparameters. The accuracy score are similar.

Which model to choose?

	precision	recall
0-10%	0.84	0.97
10%-15%	0.26	0.15
15%-20%	0.31	0.06
20%-25%	0.24	0.15
25%<=	0.48	0.52
micro avg	0.76	0.76
macro avg	0.43	0.37
weighted avg	0.70	0.76

AdaBoost

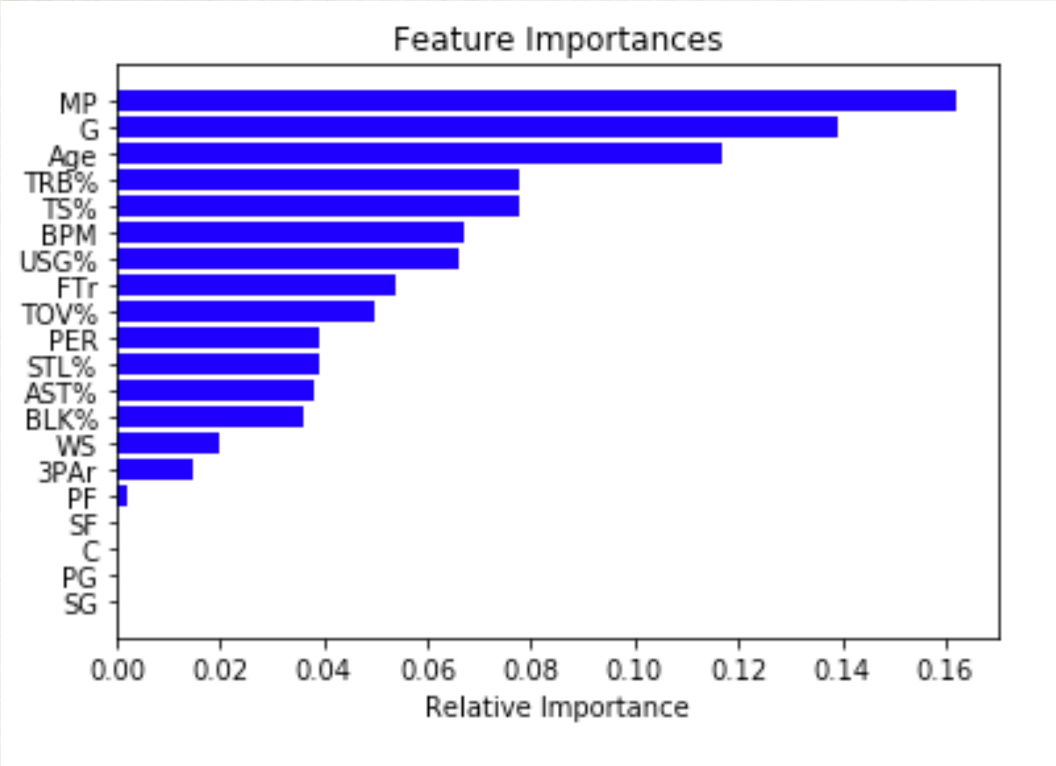
	precision	recall
0-10%	0.79	0.99
10%-15%	0.34	0.06
15%-20%	0.27	0.02
20%-25%	0.00	0.00
25%<=	0.47	0.52
micro avg	0.76	0.76
macro avg	0.38	0.32
weighted avg	0.66	0.76

Random Forest

Based on the nature of the problem, I want the end users of this model to see if that player was misclassified, meaning that the player could be overperforming/underperforming within their salary bucket. High recall and precision will be considered for model selection

AdaBoost is a winner!

Future Work



AdaBoost Feature Imporance

Minutes Played, Games Played and Age played a more important role than the other features

How to improve?

- Acquire more features such as contract information, NBA salary cap information, years of NBA experience, and basic NBA stats instead of advanced stats. This could lead to a various other projects.

Conclusion

By using NBA data like minutes played, games played, and age I was able to use AdaBoost to predict the salary percent range of an NBA player with an accuracy score of 76.4% There's much more factors that are involved that were not considered in this project, such as NBA contract data, that could make this model more perform better.