Final Project Report

Predicting NBA Salaries with Machine Learning

Group 9

Fuqi (Oren) Li, Nibin Koshy and Qiuyan Xu

McMaster University

Course: SEP786 Artificial Intelligence and Machine Learning Fundamentals

Dr. Sayyed Faridoddin Afzali

Date: July 12, 2024

Contents

1 Executive Summary 3
2 Problem Statement 3
2.1 Introduction 3
2.2 Objective 4
3 Literature Review 5
3.1 Machine Learning in NBA Analysis 5
3.2 Regularization techniques 5
3.3 Cross-Validation Technique 7
4 Discussion and Methodology 7
4.1 Data Collection 7
4.2 Data Processing 7
4.3 Model and Analysis 10
5 Result 10
5.1 Model Performance 10
5.2 Feature Importance 13
6 Conclusion and Limitation 15
References 16

Final Project Report

Predicting NBA Salaries with Machine Learning

1 Executive Summary

In this work, we aim to predict the next year's salary as a percentage of the annual salary cap of an NBA player by using 54 essential features. These statistics are used to evaluate players' performances and contributions to their teams, providing a comprehensive overview of their efficiency, shooting, rebounding, passing, and defensive capabilities. Using L1 regularization for feature selection and prediction we were able to compute the next year's salary with a high confidence level.

2 Problem Statement

In this study, our objective is to develop a predictive model that estimates the next year's salary as a percentage of the annual salary cap for NBA players. The prediction leverages 54 essential features that encompass a comprehensive range of player performance statistics, including efficiency, shooting, rebounding, passing, and defensive capabilities. By applying L1 regularization for feature selection and prediction, we aim to accurately compute the next year's salary with a high degree of confidence, thereby providing valuable insights for teams and analysts in player salary forecasting and budget planning.

2.1 Introduction

In the dynamic world of professional basketball, player salaries are a pivotal aspect of team management and overall strategy. As NBA teams operate under a structured salary cap system, accurately predicting player salaries for the upcoming season is of paramount importance. This prediction not only influences team composition and financial planning but also aids in making informed decisions regarding player acquisitions, trades, and contract negotiations. In this context, our study focuses on developing a robust predictive model that estimates the next year's salary of NBA players as a percentage of the annual salary cap, utilizing an extensive array of player performance statistics.

The NBA salary cap is designed to ensure competitive balance within the league by limiting the total amount a team can spend on player salaries. Despite this regulation, salaries can vary widely based on a multitude of factors, including a player's performance, marketability, and overall contribution to the team. Predicting these salaries accurately is a complex task, necessitating a deep understanding of the various performance metrics that contribute to a player's value.

Our approach involves leveraging 54 essential features that provide a comprehensive overview of an NBA player's performance. These features encompass a wide range of statistics, including efficiency metrics like Player Efficiency Rating (PER) and True Shooting Percentage (TS%), shooting percentages, rebounding rates, assist percentages, and defensive metrics such as Steal Percentage (STL%) and Block

Percentage (BLK%). By incorporating such a broad spectrum of data, we aim to capture the multifaceted nature of player performance and its impact on salary determinations.

A key aspect of our methodology is the use of L1 regularization, also known as Lasso regression, for feature selection and prediction. L1 regularization is particularly suited for this task as it not only aids in preventing overfitting but also performs automatic feature selection by shrinking the coefficients of less important features to zero. This results in a more interpretable model that highlights the most significant predictors of a player's future salary. The ability of L1 regularization to handle high-dimensional data makes it an ideal choice for our model, given the extensive number of features involved.

In constructing our predictive model, we first preprocess the data to ensure accuracy and consistency. This includes normalizing the features to account for differences in scale and handling any missing or anomalous values. We then split the data into training and testing sets to evaluate the performance of our model. The training set is used to fit the model, while the testing set provides an unbiased evaluation of its predictive power.

Once the model is trained, we assess its performance using metrics such as mean squared error (MSE) and R-squared (R^2) to determine its accuracy and explanatory power. A high R^2 value indicates that the model successfully captures the variability in player salaries, while a low MSE reflects precise predictions. Through iterative tuning and validation, we strive to enhance the model's performance and reliability.

The outcomes of our study have significant implications for NBA teams and analysts. An accurate salary prediction model can serve as a valuable tool for teams in managing their salary cap and making strategic decisions. It allows for more informed contract negotiations and aids in identifying undervalued or overvalued players based on their performance metrics. Additionally, this model can provide insights into the factors that most significantly influence player salaries, offering a deeper understanding of the economics of professional basketball.

In conclusion, our study aims to bridge the gap between player performance and salary prediction by employing a sophisticated statistical approach that leverages a comprehensive set of performance features. By utilizing L1 regularization for feature selection and prediction, we aim to develop a model that not only predicts next year's salary with high confidence but also enhances the interpretability and strategic value of the predictions. This endeavor contributes to the ongoing efforts to optimize team management and financial planning in the competitive landscape of the NBA.

2.2 Objective

The objective of this study is to develop a robust and accurate predictive model that estimates the next year's salary of NBA players as a percentage of the annual salary cap. This model will leverage 54 essential performance features, encompassing a wide range of player statistics such as efficiency, shooting, rebounding, passing, and defensive capabilities. By applying L1 regularization for feature selection and prediction, we aim to ensure the model's interpretability and prevent overfitting, thereby achieving high-confidence salary predictions. Ultimately, this model will serve as a valuable tool for NBA

teams and analysts in managing salary caps, making informed contract negotiations, and understanding the key performance factors influencing player salaries.

3 Literature Review

3.1 Machine Learning in NBA Analysis

In the fast-evolving landscape of the National Basketball Association (NBA), data analytics has ignited a transformative revolution. As a popular game, NBA has wealth data which can facilitate the data-driven analysis about various facets of NBA, such as forecasting player performances, predicting game results and points, optimization in fantasy sports (Papageorgiou et al., 2024). Furthermore, as one of the biggest professional sports leagues, NBA has significant commercial value. Therefore, the prediction of athlete salaries using machine learning and statistical models has been a focal point recently. Several research aimed to find the model for predicting salary based on the performance (Pastorello, 2023; Wang, 2024; Welsh, 2024). However, there are other aspects might impact the salary. For example, Lee et al. (2023) discussed the social media impact on player salary in NBA. Also, some researcher indicated that the salary is not in accordance with the performance on court (Özbalta et al, 2022). It is noticed that sometimes the team cannot pay as much as they want to sign the best players, because the NBA operates under a complex set of rules and regulations that aim to maintain competitive balance among teams (Akabas, 2024). One of the regulations is the salary cap, which will be published every season, after considering the economic environment and the sports market. The salary cap's evolution can reflect the inflation and the market value of the game. To account the cap's evolving nature, Pastorello (2023) used the percentage of the cap as the target rather than the actual salary amount, ensuring the results are not influenced by temporal changes and remain applicable even when analyzing past seasons.

Among the research, Linear regression remains a cornerstone in the prediction of NBA salaries due to its simplicity and interpretability. This statistical method models the relationship between a dependent variable (salary) and independent variables (performance metrics). For example, Feng et al. (2023) applied multivariate linear regression to predict NBA player salaries and compared different regularization methods, such as Lasso, Ridge, and Elastic Net.

3.2 Regularization techniques

Regularization techniques are pivotal in machine learning for improving model performance and generalizability, particularly when dealing with high-dimensional data. These methods introduce a penalty to the model for having large coefficients, thereby discouraging overfitting and simplifying the model. There are three key types of regularization: Lasso Regression, Ridge Regression, and Elastic Net (Bhadauriya, 2021).

Lasso Regularization (L1 Regularization) (Least Absolute Shrinkage and Selection Operator) performs both variable selection and regularization to enhance the prediction accuracy and interpretability of the statistical model it produces. It accomplishes this by introducing a penalty term to the regression model,

which can shrink some of the coefficients to zero, effectively selecting a simpler model that retains only the most significant features and preventing overfitting.

The modified loss function in Lasso regression is:

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j| \;=\mathrm{RSS}+\lambda\sum_{j=1}^{p}|\beta_j|$$

Here: λ is the tunable parameter that controls the degree of shrinkage.

Ridge Regularization (L2 Regularization) incorporates a regularization term to address multicollinearity among predictors and prevent overfitting. Unlike Lasso, Ridge does not perform feature selection by shrinking coefficients to zero; instead, it reduces the magnitude of all coefficients by adding a penalty term to the regression model.

The modified loss function in Ridge regression is:

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}\beta_j^2=\mathrm{RSS}+\lambda\sum_{j=1}^{p}\beta_j^2$$

Here: λ is the tunable parameter that controls the degree of shrinkage.

The inclusion of the L2 penalty term ensures that large coefficients are penalized more heavily, thereby shrinking them. This helps in maintaining all the features in the model but with reduced effect sizes, leading to a more generalized model.

Elastic Net combines the properties of both Ridge and Lasso regularization, incorporating penalties from both L1 and L2 regularization. This model is particularly useful when there are multiple correlated predictors, as it can handle the complexity better by balancing the trade-offs between Ridge and Lasso.

The modified loss function in Elastic Net is:

$$\frac{\sum_{i=1}^{n}(y_i-x_i^J\hat{\beta})^2}{2n}+\lambda\left(\frac{1-\alpha}{2}\sum_{j=1}^{m}\hat{\beta}_j^2+\alpha\sum_{j=1}^{m}|\hat{\beta}_j|\right)$$

Here: λ is the tunable parameter that controls the degree of shrinkage, α is the percentage of L1 Regularization.

The L1 component encourages sparsity by shrinking some coefficients to zero, effectively performing feature selection. The L2 component helps distribute the load among correlated features and stabilizes the model. This combination makes Elastic Net a versatile and powerful tool for dealing with complex datasets with numerous predictors, especially when predictors are highly correlated or when the number of predictors exceeds the number of observations.

3.3 Cross-Validation Technique

Cross-validation is an advanced technique in machine learning for evaluating a model's generalization capability to unseen data (Brownlee, 2023). It splits the original dataset into separate parts for training the model and tests its performance. This method helps in identifying and mitigating overfitting, ensuring that the model not only fits well to the training data but also performs accurately on new data.

One of the most common cross-validation techniques is k-fold cross-validation. The dataset is divided into k subsets. Then the model will be trained based on k-1 of the folds and tested on the remaining one. This process is repeated k times and each subset is used as the test set once. The performance results from each fold are averaged to provide a more reliable evaluation metric for the model. This technique is a robust way to assess the model's performance by reducing the variability associated with a single train-test split.

4 Discussion and Methodology

4.1 Data Collection

The data sources of the project are Kaggle.com and the website of Basketball Reference. These two websites have reliable and latest official data of the NBA. The data for this project mainly includes performance data of NBA players, Salary data, and Salary Cap of the team.

Performance data includes multiple dimensions of player information, including:

- Personal information: Name, Age, Team, and Position.
- Basic statistics: Games played (G), games started (GS), and minutes played (MP).
- Shooting metric: Points (PTS), field goals made (FG), field goals attempted (FGA), and field goal percentage (FG%) were used alongside three-point field goals made (3P), three-point field goals attempted (3PA), and three-point field goal percentage (3P%).
- Defensive metrics: Steals (STL), blocks (BLK), and personal fouls (PF).
- Advanced metrics: Player Efficiency Rating (PER), True Shooting Percentage (TS%), Three-Point Attempt Rate (3PAr), Free Throw Rate (FTr), Offensive Rebound Percentage (ORB%), Defensive Rebound Percentage (DRB%), Total Rebound Percentage (TRB%), etc.

Salary information is from 'NBA Salaries (1990-2021).csv', which includes a player's salary for that year. The salary cap data, which limits the total amount a team can spend on player salaries in the NBA, is obtained from 'Salary Cap.csv.'

This project will analyze and process the data in these three datasets, to predict the salary of next year based on the performance and salary of players this year.

4.2 Data Processing

4.2.1 Combine Dataset

Since this project involves data from performance, salary, and salary cap, which are not stored in a single database, it is necessary to merge these datasets before any further operations.

- **Loading Data**

```
seasons_df = pd.read_csv("/Users/oren/Desktop/ML_Project/Seasons_Stats.csv", encoding='latin-1')
salaries_df = pd.read_csv("/Users/oren/Desktop/ML_Project/NBA Salaries(1990-2021).csv")
cap_df = pd.read_csv("/Users/oren/Desktop/ML_Project/Salary Cap.csv")
```

- **Data Cleaning**

Due to differences in cell attributes from different tables, trimming and converting are required to maintain data consistency.

```
seasons_df['Player'] = seasons_df['Player'].str.strip()
salaries_df['Player'] = salaries_df['Player'].str.strip()
seasons_df['Year'] = seasons_df['Year'].astype(int)
salaries_df['Year'] = salaries_df['Year'].astype(int)
cap_df['Year'] = cap_df['Year'].astype(int)
```

- **Merging Datasets**

After cleaning, the relevant columns can be unified and merged into a single table.

```
salaries_relevant_columns = salaries_df[['Player', 'Year', 'salary', 'inflationAdjSalary']]
merged_df = pd.merge(seasons_df, salaries_relevant_columns, on=['Player', 'Year'], how='left')
merged_df.dropna(subset=['salary'], inplace=True)
cap_relevant_columns = cap_df[['Year', 'SalaryCap']]
merged1_df = pd.merge(merged_df, cap_relevant_columns, on=['Year'], how='left')
merged1_df.dropna(subset=['SalaryCap'], inplace=True)
```

4.2.2 Imputing Missing Data

In the project, we found that the three-point percentage of some players was empty, which means these players did not shoot a three-point shot in the year, so we removed these data.

```
df = df.dropna()
```

4.2.3 Feature Creation

For better analysis and modeling, we created a new feature based on the original data.

Player salary as a percentage of the salary cap (percentageSalary): The salary cap of NBA teams varies significantly from year to year, so the actual value of salary varies from year to year. By calculating the percentage of the salary cap, players' salary data can be better standardized, making the data comparable between different years. It can also improve the interpretability of the model.

```
columns_to_clean = ['salary', 'SalaryCap']
for col in columns_to_clean:
   df[col] = df[col].astype(str).str.replace('$', '').str.replace(',', '')
df['salary'] = df['salary'].astype(int)
df['SalaryCap'] = df['SalaryCap'].astype(int)
```

```
df['percentageSalary'] = df['salary'] / df['SalaryCap']
df['Next_Year_Percentage'] = df.groupby('Player')['percentageSalary'].shift(-1)
df.dropna(subset=['Next_Year_Percentage'], inplace=True)
```

4.2.4 Feature Extraction or Dropping

To ensure the reliability of the data and the accuracy of the analysis, we deleted players who played less than 20 times in a year and reserved only the total performance for players who played for multiple teams in the same season. We also removed the player column to prevent one hot from causing a high-latitude disaster. By refining the dataset in this manner, we ensured that only the most relevant player records were retained for further analysis and modeling.

```
def filter_players(group):
    if len(group) > 1:
        return group[group['Tm'] == 'TOT']
    else:
        return group
filtered_df = df.groupby(['Year', 'Player']).apply(filter_players).reset_index(drop=True)
df = df[df['G'] >= 20]
df = df.drop('Player',axis=1)
```

4.2.5 One-hot Encoding

Since the data contains some string variables, such as Pos and Team, these features are important for predicting the result, so these features need to be one-hot encoding.

```
df_objs = pd.get_dummies(df[['Tm', 'Pos']], drop_first=False, dtype=int)
df_nums = df.select_dtypes(exclude='object')
df = pd.concat([df_nums,df_objs],axis=1)
```

Before one-hot encoding, the table had 54 features, and after one-hot encoding, the number of features increased to 104.

	Year	Age	G	GS	MP	FG	FGA	FG%	3P	3PA	...	Pos_PG-SF	Pos_PG-SG	Pos_SF	Pos_SF-C	Pos_SF-PF	Pos_SF-SG	Pos_SG	Pos_SG-PF	Pos_SG-PG
0	1990	26.0	82	82.0	2709.0	385	806	0.478	13.0	46.0	...	0	0	0	0	0	0	0	0	0
2	1990	27.0	81	81.0	2599.0	476	946	0.503	4.0	26.0	...	0	0	0	0	0	0	1	0	0
4	1990	28.0	77	4.0	1727.0	356	721	0.494	0.0	7.0	...	0	0	0	0	0	0	0	0	0
5	1990	25.0	76	66.0	2426.0	484	940	0.515	0.0	2.0	...	0	0	0	0	0	0	0	0	0
6	1990	24.0	53	10.0	575.0	55	142	0.387	1.0	4.0	...	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
8677	2021	27.0	53	16.0	906.0	115	182	0.632	1.0	11.0	...	0	0	0	0	0	0	0	0	0
8678	2021	26.0	50	4.0	723.0	79	180	0.439	36.0	90.0	...	0	0	1	0	0	0	0	0	0
8679	2021	25.0	58	58.0	2034.0	569	1123	0.507	200.0	477.0	...	0	0	0	0	0	0	1	0	0
8680	2021	20.0	42	1.0	397.0	50	104	0.481	24.0	59.0	...	0	0	0	0	0	0	0	0	0
8681	2021	20.0	61	61.0	2026.0	634	1037	0.611	10.0	34.0	...	0	0	0	0	0	0	0	0	0

8319 rows × 104 columns

Fig. 1 Dataset after on-hot encoding

4.3 Model and Analysis

For feature selection, we used player performance metrics and the percentage of the current year's salary to the salary cap as X, and the percentage of next year's salary to the salary cap as y. These features are then standardized.

```
X = df.drop('Next_Year_Percentage', axis=1)
y = df[['Next_Year_Percentage']]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=42)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Using the percentage of next year's salary to the salary cap as a target variable can standardize salary data, reflect the relative value of players and resource allocation, enhance the explanatory power of the model, and ensure comparability across years and teams to predict future salary levels more accurately.

We experimented with three regression models: Lasso (L1), Ridge (L2), and Elastic Net. By comparing the prediction and training results of these models, we selected the best-performing one.

5 Result

5.1 Model Performance

5.1.1 Lasso Regression

Lasso Regression introduces an absolute value term penalty that makes some coefficients zero, thus enabling feature selection. During the experiment, we tested the performance of different Alpha.

- Alpha = 0: There is no regularization, and the model is completely dependent on the training of the data. This can lead to overfitting because no features are penalized or excluded.
- Alpha = 0.0004906: The best alpha value found by cross-validation. This means that at this alpha value, the model has reached the optimal balance point, which can effectively reduce overfitting while maintaining high prediction accuracy.

We use several alpha values and cross-validate to select the best alpha value. The specific code is as follows.

```
from sklearn.linear_model import Lasso, LassoCV
lasso_cv = LassoCV(cv=5)
lasso_cv.fit(X_train_scaled, y_train)
optimal_alpha = lasso_cv.alpha_

lasso1 = Lasso(alpha=optimal_alpha)
lasso1.fit(X_train_scaled, y_train)

y_pred_lasso1 = lasso1.predict(X_test_scaled)
y_train_lasso1 = lasso1.predict(X_train_scaled)
```

Through experiments, the results are shown in the following table.

Table 1. Results of Different Alpha Values for Lasso Regularization

Alpha	MSE Test	RMSE Test	R2 Test	MSE Train	RMSE Train	R2 Train
0	0.001439	0.037928	0.79269	0.001452	0.038106	0.801389
0.0004906	0.00144	0.037946	0.792491	0.001471	0.038358	0.798755

As can be seen from the table above, using the best alpha value (0.0004906) can reduce overfitting while maintaining high prediction accuracy. Although the performance on the training data decreased slightly, the performance on the test data improved, indicating that the model had better generalization ability.

5.1.2 Ridge Regression

Ridge Regression introduces a penalty of a squared term, narrowing some coefficients but not reducing them to zero. During the experiment, we tested the performance of different Alpha.

- Alpha = 0: There is no regularization, and the model is completely dependent on the training of the data. This can lead to overfitting because no coefficients are reduced.
- Alpha = 10: The best alpha value found by cross-validation. At this alpha value, the model reduces overfitting by imposing the same penalty on all coefficients, while maintaining good predictive power.

We use many different alpha values and cross-validate to select the best alpha value. The specific code is as follows.

```
from sklearn.linear_model import RidgeCV
alphas = [0.1, 1.0, 10.0, 100.0, 200.0]
ridge_cv = RidgeCV(alphas=alphas, cv=5)
ridge_cv.fit(X_train_scaled, y_train)
optimal_alpha_ridge = ridge_cv.alpha_

Ridge1 = Ridge(alpha= ridge_cv.alpha_)
Ridge1.fit(X_train_scaled, y_train)
y_pred_Ridge1 = Ridge1.predict(X_test_scaled)
y_train_Ridge1 = Ridge1.predict(X_train_scaled)
```

Through experiments, the specific results are shown in the following table.

Table 2. Results of Different Alpha Value for Ridge Regularization

Alpha	MSE Test	RMSE Test	R2 Test	MSE Train	RMSE Train	R2 Train
0	0.001439	0.037931	0.792655	0.001452	0.038105	0.801401
10	0.001438	0.037921	0.792764	0.001453	0.038114	0.801314

From the table above, using the optimal alpha value (10) can reduce overfitting while maintaining good predictive performance. Although there is a slight decrease in the performance of the training data, the performance of the test data improves.

5.1.3 Elastic Net

Elastic Net combines the advantages of L1 and L2 to flexibly adjust the parameters of the model by introducing two hyperparameters, alpha and l1_ratio. During the experiment, we tested the performance of different Alpha and L1_ratio.

- Alpha = 0.01, L1_ratio = 0.01: The best combination of parameters found by cross-validation. Under this combination, the model makes use of the feature selection ability of L1 regularization and the stability of L2 regularization to achieve the best prediction effect.
- Other combinations: Under other alpha and l1_ratio combinations, the model's predictions were poor, indicating that these combinations were not optimally balanced.

We used a variety of different alpha and l1_ratio values and cross-validated to select the best combination of parameters. The specific code is as follows.

```
from sklearn.model_selection import train_test_split, cross_val_score
alphas = [0.01, 1.0, 10.0]
l1_ratios = [0.01, 0.5, 0.9]
best_alpha = None
best_l1_ratio = None
best_score = -np.inf

for alpha in alphas:
  for l1_ratio in l1_ratios:
    elastic_net = ElasticNet(alpha=alpha, l1_ratio=l1_ratio, random_state=42)
    scores = cross_val_score(elastic_net, X_train_scaled, y_train, cv=5, scoring='r2')
    mean_score = np.mean(scores)
    if mean_score > best_score:
      best_score = mean_score
      best_alpha = alpha
      best_l1_ratio = l1_ratio
```

Through experiments, the specific results are shown in the following table.

Table 3. Results of Different Parameter Value for Elastic Net

Alpha	L1_ratio	R2 Test
0.01	0.01	0.792891
0.01	0.5	0.784803
0.01	0.9	0.774246
1	0.01	0.64036
1	0.5	-0.001442
1	0.9	-0.001442
10	0.01	-0.001442
10	0.5	-0.001442
10	0.9	-0.001442

As can be seen from the above table, the optimal combination of parameters (Alpha = 0.01, L1_ratio = 0.01) can significantly improve the prediction effect of the model. The combination utilizes the feature selection ability of L1 regularization and the stability of L2 regularization to achieve the best prediction effect.

5.1.4 Comparison

Through the above comparison, we find that the values of mean square error (MSE), root mean square error (RMSE), and R2 scores for the training and test datasets were very similar across all three models.

Table 4. Results Comparison of Different Methods

Model	MSE Test	RMSE Test	R2 Test	MSE Train	RMSE Train	R2 Train
Lasso	0.00144	0.037946	0.792491	0.001471	0.038358	0.798755
Ridge	0.001438	0.037921	0.792764	0.001453	0.038114	0.801314
Elastic Net	0.001437	0.03791	0.792891	0.001459	0.038191	0.800506

However, the Lasso model uses significantly fewer features, which can be advantageous.

Table 5. Features Selected by Different Methods

Model	Feature
Lasso	31
Ridge	103
Elastic Net	77

Overall, while the performance metrics for Lasso, Ridge, and Elastic Net are very similar, the Lasso model has a smaller number of features, and the model is more user-friendly.

5.2 Feature Importance

To assess the most important features, feature selection was done using the LassoCV method. LassoCV for feature selection is a powerful technique in machine learning that leverages L1 regularization to enhance model accuracy and interpretability. LassoCV (Least Absolute Shrinkage and Selection Operator with Cross-Validation) automatically selects the most important features by penalizing the absolute size of the regression coefficients. As the penalty increases, less important feature coefficients shrink to zero, effectively performing feature selection. This method not only helps in reducing the complexity of the model by keeping only the significant features but also improves generalization by preventing overfitting. The cross-validation aspect ensures that the model's performance is robust and not overly dependent on a particular subset of the data, making LassoCV a reliable choice for creating parsimonious and high-performing predictive models.

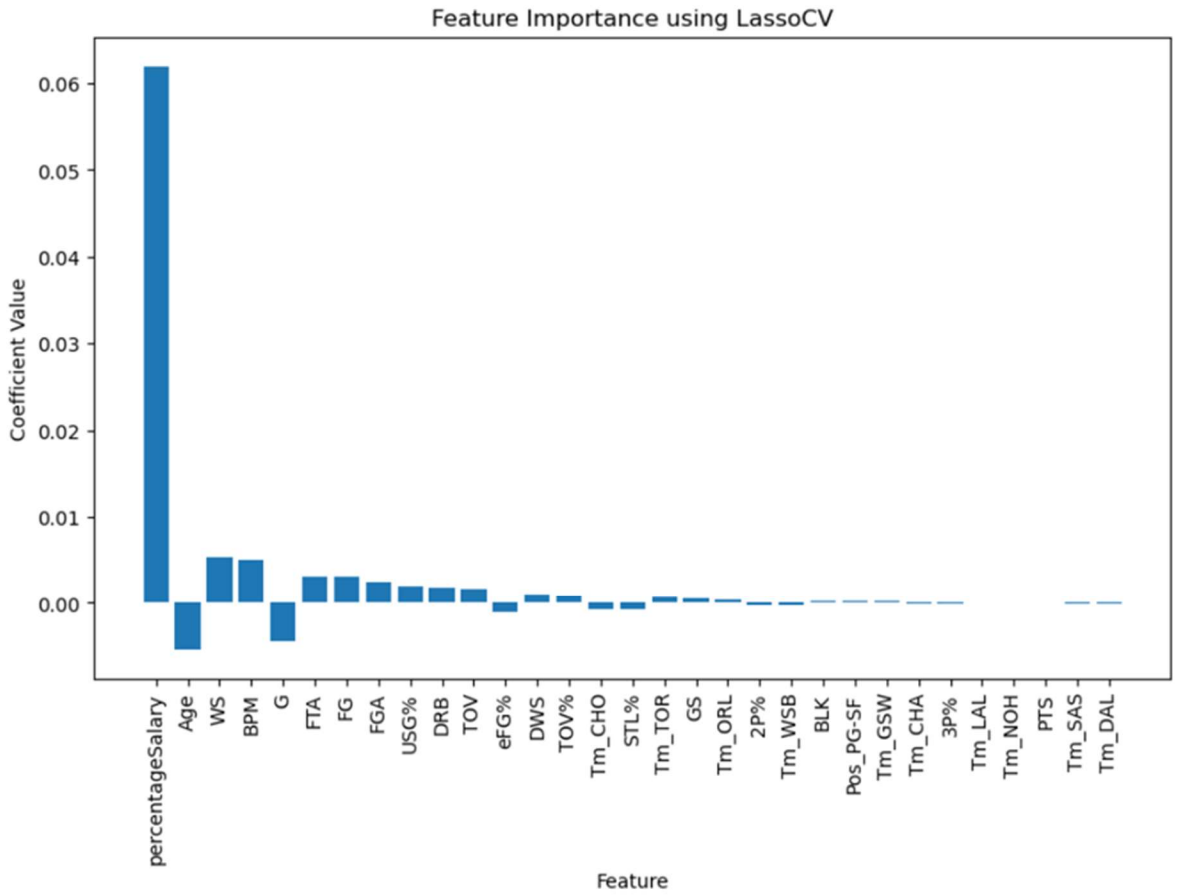

Fig. 2 Feature selection using LassoCV

In the present model we were able to reduce the number of features to 31 through the LassoCV method and the most important features to determine an NBA player's next year salary was the percentage salary, age, win share, box plus-minus, games, and free throw attempts (Figure 2).

Percentage Salary: This refers to the percentage of the team's salary cap that a player's salary represents. It reflects the player's contractual value relative to the team's financial constraints and salary cap management.

Age: The player's age is a significant factor as it correlates with experience, physical condition, and potential longevity in their career. Younger players may command higher salaries due to potential growth, while older players may negotiate based on their experience and leadership.

Win Shares: Win Shares quantify the number of wins a player contributes to their team through both offensive and defensive performance. Players with higher Win Shares are typically considered more impactful and may command higher salaries.

Box Plus-Minus (BPM): BPM measures a player's overall impact per 100 possessions relative to an average player. It combines offensive and defensive contributions, providing insight into a player's comprehensive influence on team performance.

Games: The number of games a player participates in reflects their availability and durability. Players who consistently play a high number of games demonstrate reliability and may negotiate higher salaries based on their availability to contribute to the team.

Free Throw Attempts: This statistic indicates how often a player gets to the free-throw line, which reflects their ability to draw fouls and convert scoring opportunities. Players proficient in free throw attempts often demonstrate offensive aggression and efficiency, influencing their market value.

These features collectively provide a nuanced view of a player's performance, experience, impact on team success, and marketability which are all factors that NBA teams consider when determining player salaries and contract negotiations.

6 Conclusion and Limitation

Predicting salaries in sports is complex due to the wide range of variables and external economic changes. This project aimed to predict the next year's percentage of the salary cap by considering the current salary and the salary cap of that year. By incorporating the salary cap, the model accounts for inflation and market value, resulting in predictions that better reflect the relative value within the NBA league. The results from the regressions indicate that players' current percentage of the salary cap, age, win shares, box plus-minus, games played, free throw attempts are significant factors influencing the next year's percentage of the cap or player value.

The model was analyzed using Lasso, Ridge, and Elastic net regression and it was found that all the models predicted with elevated level of accuracy with identical R2 scores with Elastic net performing marginally better than the other two models. However, Lasso was able to reduce the number of features from 103 to 31 while Elastic net reduced the number of features to 77, this reduces the complexity of the problem and reduces the computational load on the system.

Some players played for multiple teams in one season, which has resulted in their cumulative data being recorded, potentially affecting the quality of the dataset. It is also important to note that the most significant feature from the dataset is the player's current percentage salary. This makes it challenging to assess the percentage salary for players with no prior history in the NBA.

While the model improves by including the salary cap, other significant factors remain unaccounted for. For instance, the impact of players' presence on social media, their experience, and their development potential can significantly influence their value. Additionally, unpredictable factors such as injuries can affect actual performance and, consequently, the accuracy of predictions.

References

Akabas, L. (2024, July 5). HOW DO NBA CONTRACTS AND THE SALARY CAP WORK? *Sportico.com*. https://www.sportico.com/feature/nba-salaries-explained-salary-cap-1234786618/

Brownlee, J. (2023). *A gentle introduction to K-Fold Cross-Validation*. Machine Learning Mastery. https://machinelearningmastery.com/k-fold-cross-validation/

Bhadauriya, R. (2021). Lasso ,Ridge & Elastic Net Regression: A Complete Understanding (2021). *Medium*. https://medium.com/@creatrohit9/lasso-ridge-elastic-net-regression-a-complete-understanding-2021-b335d9e8ca3

Feng, X., Wang, Y., & Xiong, T. (2023). NBA Player Salary Analysis based on Multivariate Regression Analysis. *Highlights in Science, Engineering and Technology*, *49*, 157-166.

Özbalta, E., Yavuz, M., & Kaya, T. (2022). National Basketball Association Player Salary Prediction Using Supervised Machine Learning Methods. *In Intelligent and Fuzzy Techniques for Emerging Conditions and Digital Transformation Proceedings of the INFUS 2021 Conference, held August 24-26, 2021*. Volume 2 / (Vol. 308, pp. 189–196). Springer International Publishing: https://doi.org/10.1007/978-3-030-85577-2_22

Papadaki, I., & Tsagris, M. (2022). Are NBA Players' Salaries in Accordance with Their Performance on Court? In *Advances in Econometrics, Operational Research, Data Science and Actuarial Studies: Techniques and Theories* (pp. 405-428). Cham: Springer International Publishing.

Papageorgiou, G., Sarlis, V., & Tjortjis, C. (2024). An innovative method for accurate NBA player performance forecasting and line-up optimization in daily fantasy sports. *International Journal of Data Science and Analytics*. https://doi.org/10.1007/s41060-024-00523-y

Pastorello, G. (2023, September 9). Predicting NBA Salaries with Machine Learning | Gabriel Pastorello | Towards Data Science. *Medium*. https://towardsdatascience.com/predicting-nba-salaries-with-machine-learning-ed68b6f75566

Wang, D. (2024, March 14). Predictive modeling of NBA player salaries using machine learning. *Medium*. https://medium.com/@dwang22/predictive-modeling-of-nba-player-salaries-using-machine-learning-techniques-36b6ab71d11b

Lee, C. C., Zhang, R., Lomotey, R., Watkins, J., & Huang, Y. (2023). Examining the impact of social media following on player salary in the national basketball association: a multivariate statistical analysis. *Journal of Applied Business and Economics*, 25(1).

Welsh, J. (2024, January 28). *Predicting NBA Salaries w/ Gradient Boosting*. Kaggle. https://www.kaggle.com/code/jamiewelsh2/predicting-nba-salaries-w-gradient-boosting