Aaron Zachariah

Final Project Report

Overview

For my final project, I created a regression analysis on NBA win percentages. The goal of this project was to collect and curate my own basketball data from basketball-reference [1] and use that data to fit various regression models, evaluate their performance and use them to make way-too-early predictions for the final standings for this NBA season. This project will be using data preprocessing techniques mentioned in class, like normalization. For this project, I will also be using three different regression models based on methods discussed in class – the standards Ordinary Least Squares Regression, Support Vector Regression, and Random Forest Regression. This project will also require proper evaluation, so we can interpret the results and understand the success and/or failures of the methods. All of these techniques have been introduced in class and will be applied during this project.

Data Collection and Preprocessing

Data collection was one of the key pieces of this project, and arguably the most difficult/time consuming portion to implement. One of the key motivations for this project is to not use a pre-made dataset, but to curate one ourselves. To do this, I implemented web scraping tools to parse the website basketball-reference, which has season data since the inception of the NBA. I focus my efforts on collecting data from the 2000 season and later, as some older seasons may have missing data due to less tracking data at the time. I also focus on collecting team-based data, as that provides a more holistic measure of how a team is performing, without having to worry about who is on the team or who is currently playing. This is important because players can move in and out of the rotation, and sometimes players can be out for injury or mid-season trades. I scraped and parsed the following tables from basketball-reference:

- Season Standings
- Per Game Stats
- Per 100 Possession Stats
- Shooting Stats
- Advances Stats

Once the data is collected from basketball-reference, the next step is to decide what features to use when fitting a regression model. Each table has several columns of various metrics for all the teams in that particular season. However, not every column will have data that will be useful in predicting win percentage. Thus, we need to narrow down the columns and transform features to be good predictors for our response variable.

To do this, there are some important considerations to make. First, whatever features chosen must be invariant to the era of basketball that the data is drawn from. This means that the values of the features must be uninfluenced by the season it originates. A good example of this is 3-point percentage. In the 90s and early 2000s, no teams in the NBA utilized the 3-point shot that much. Thus, even championship winning teams were not necessarily good 3-point shooting teams. However, in the last decade or so, there's been a massive spike in the integration of the 3-point shot into the NBA offense. These days, to be a championship team it is nearly essential to be a competent 3-point shooting team. This means that 3-point percentage is not invariant to the era. While it may be a decent indicator of team success these days, it wasn't as much in days past. Thus it is not a good metric to consider when learning a model trained on many seasons of data.

Another important consideration is the issue of scale. The same metric may be invariant to era, but still have different scales from season to season. This can be because of a variety of factors, like relative parity or offensive/defensive efficiencies. For example, one important metric (that I end up using) is the Simple Rating System (SRS). This metric is a rating based off the margin of victory and strength of schedule. Now consider two great teams in two different seasons, which have both won 65/82 games. In one season, there is a lack of parity in the league, and that's team's SRS is lower because of the lower strength of schedule. In the other season, the league is more balanced and that good team's SRS is higher because they have the higher strength of schedule. Then, even if these two teams have the same win%, they have different SRS values because of the difference in parity between the two seasons. I attempt to address this problem through normalization (more on this later).

For my project, I ended up preprocessing several features and transforming them into a set of three metrics, which will be the predictor variable for this task. The three metrics are: Four-Factor Score, SRS, and Net Rating.

Four Factor Score is the most complex of the three, and it involves 8 total metrics to calculate it. First, I will introduce the idea of the four factors.

Four Factors: The key factors which influence winning a basketball game

- Shooting The Ball (Effective Field Goal %)
- Taking Care of the Ball (Turnover %)
- Offensive Rebounding %
- Getting to the Foul Line (Free Throw Rate)

These four factors can be extended to both offensive and defensive possessions, meaning there are Offensive and Defensive Four Factors (8 total factors). The four defensive factors are conceptually the same as the offensive, but Offensive Rebounding % becomes Defensive Rebounding %, and the rest of the metrics are based on the opponents.

Even though the four factors all contribute to winning ball games, they are not equal in value. According to NBA Stuffer [2], sports analyst and statistician Dean Oliver has found the relative contributions as follows:

1. Shooting (40%)
2. Turnovers (25%)
3. Rebounding (20%)
4. Free Throws (15%)

Using these relative contributions to winning, we can design an Offensive and Defensive Four Factor Score with the following formulas:

$$Offensive\ FourFactor\ Score = 0.40 * eFG\% - 0.25 * TOV\% + 0.20 * ORB\% + 0.15 * FT/FGA\%$$

$$Defensive\ FourFactor\ Score = -0.40 * eFG\% + 0.25 * TOV\% + 0.20 * DRB\% - 0.15 * FT/FGA\%$$

As mentioned, the values for the offensive score are the team's, and the values for the defensive score are the opponent's. The score is designed such that the higher it is, the higher the win% should be. We can then use these scores to produce a net score, which will be the final Four-Factor Score. It is defined as follows

$$Four\ Factor\ Score = Offensive\ FourFactor + Defensive\ FourFactor$$

To illustrate the transformations of the four factors, and how they correlate to winning, see the figures below. Figures 1 and 2 show the individual four factors plotted against win %. Figures 3 and 4 show the Offensive and Defensive Four Factor score vs win % respectively. Figure 5 shows the final Four Factor score vs win %.

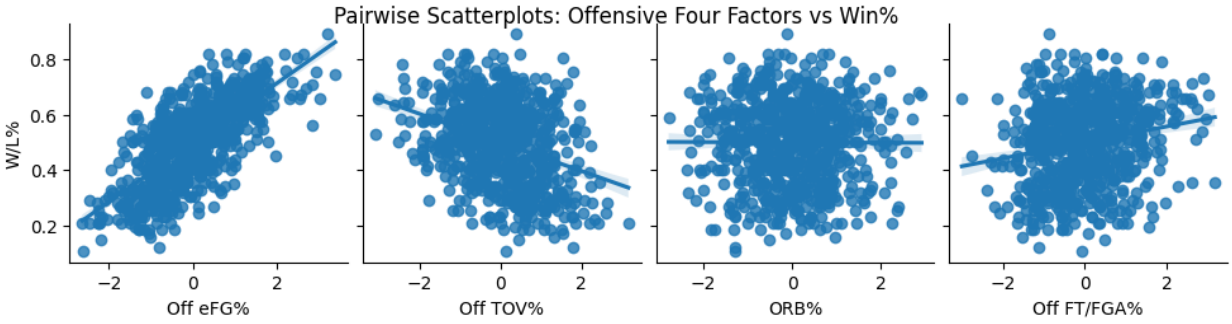

Figure 1: Offensive Four Factors vs Win %

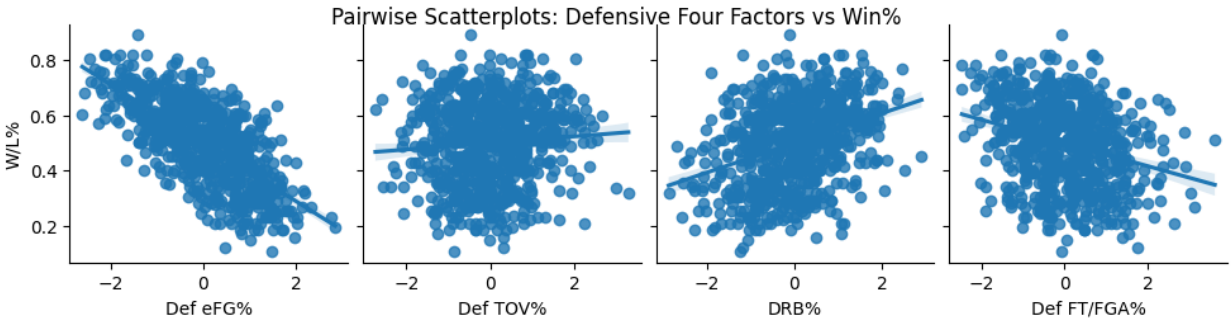

Figure 2: Defensive Four Factors vs Win %

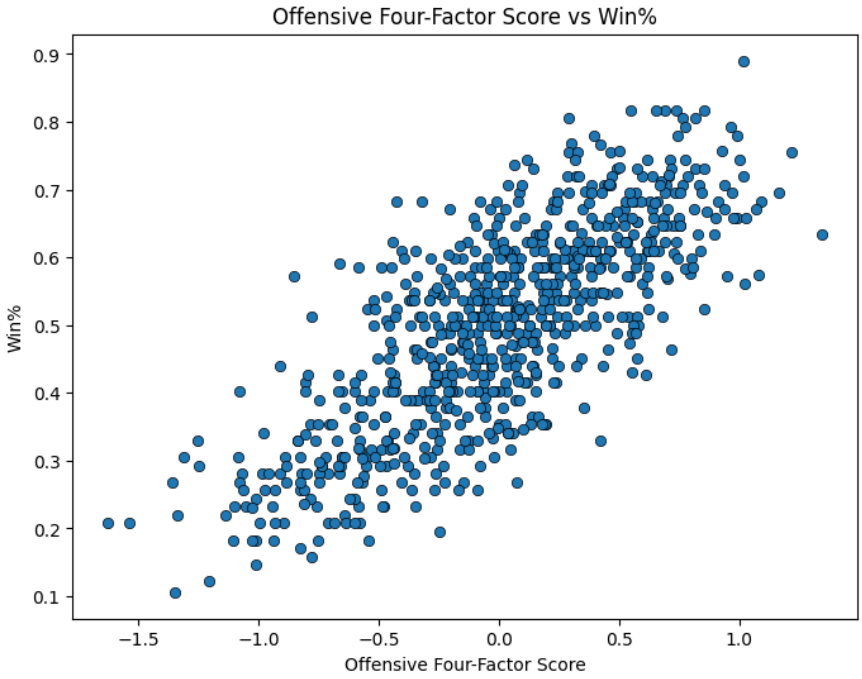

Figure 3: Offensive Four Factor Score vs Win %

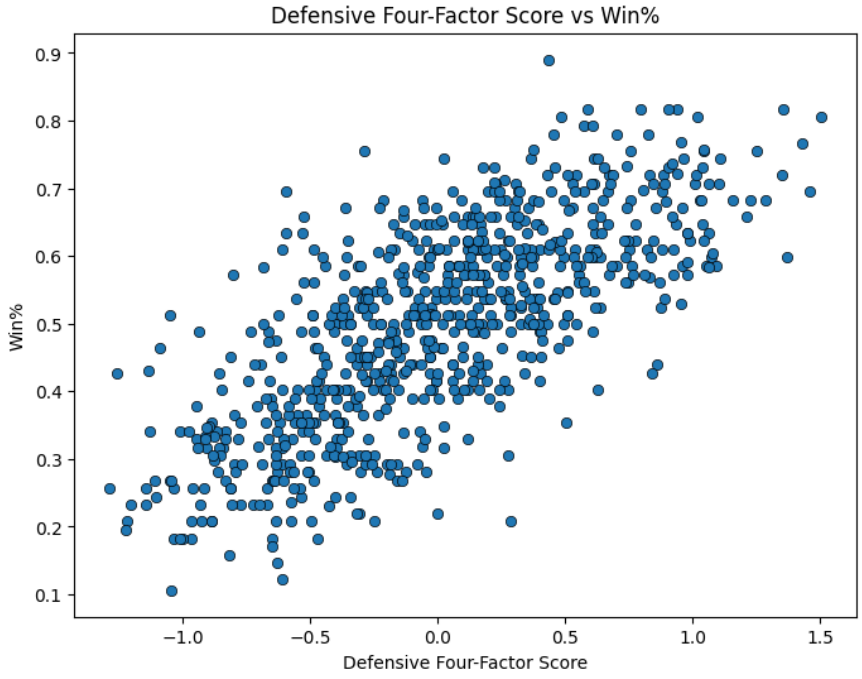

Figure 4: Defensive Four Factor Score vs Win %

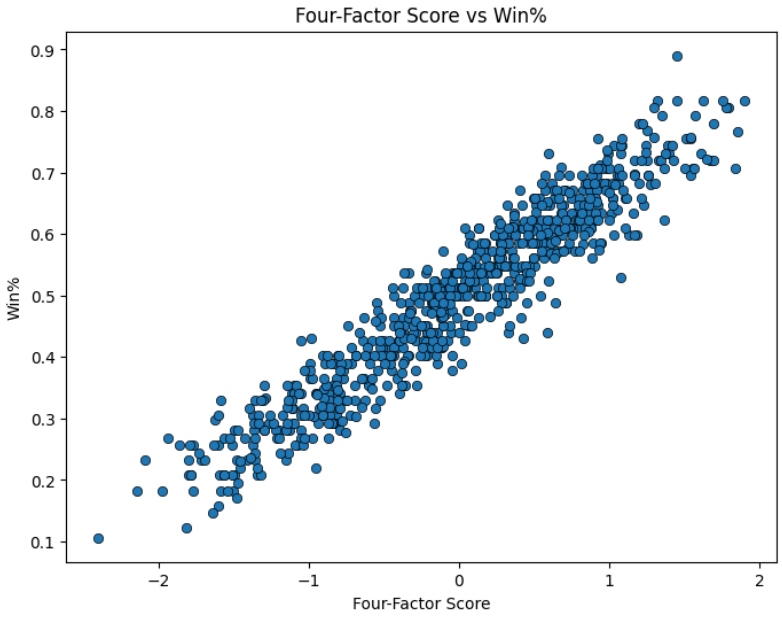

Figure 5: Four Factor Score vs Win %

Getting the Simple Rating System (SRS) and Net Rating (NRtg) metrics are thankfully much easier, as they are already available from the parsed basketball-reference data. The Simple Rating System is a rating that quantifies a team based on their average margin of victory and their strength of schedule. The Net Rating is the difference between a team's offensive and defensive efficiency. For both metrics, higher is better. As hinted earlier, these metrics can be susceptible to scaling issues, where two teams in two different seasons with the same record may have relatively large differences in their ratings. To address this, I implement a zscore-like normalization to the data for each season. Z score normalization is a technique which can rescale data to have a mean of 0 and a standard deviation of 1. This can be useful for getting data with different scales to all have a uniform scale. Z score normalization is as follows:

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

We don't need to do the full z score normalization, since SRS and Net Rating are both centered around 0 already. Thus, to normalize it I simply divide by the standard deviation of the data for that season. This will uniformly scale each season to have a standard deviation of 1. Based on figures 6 and 7, SRS and Net Rating both have strong correlations to win %.

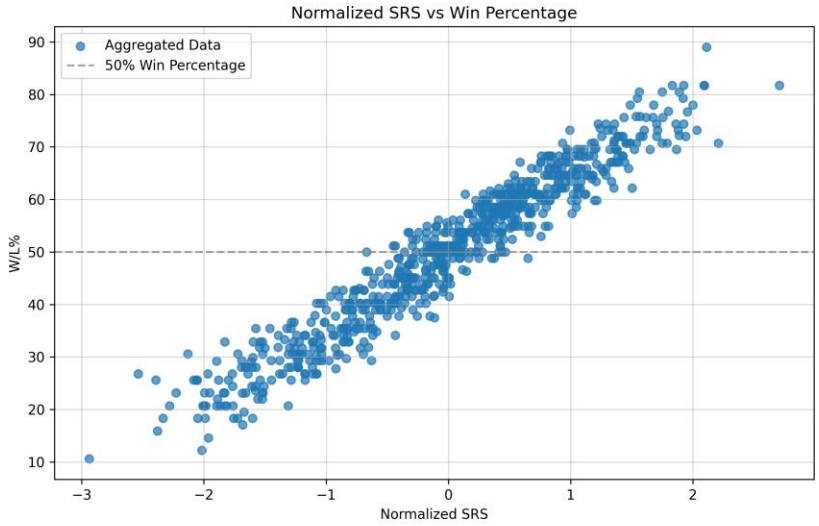

Figure 6: SRS (normalized) vs Win %

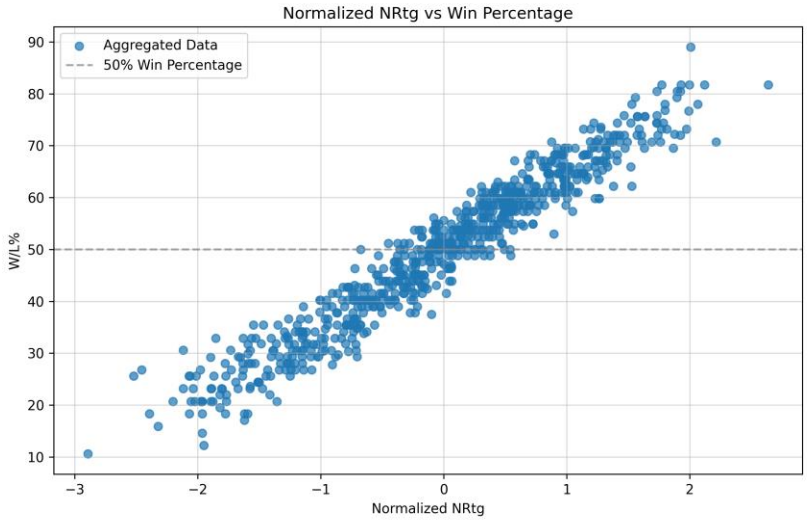

Figure 7: Net Rating (normalized) vs Win %

The Four-Factor Score, SRS and Net Rating serve as the three input features to all the regression models used in the next section. As demonstrated, these metrics show a strong correlation to winning percentage and are also robust to some of the potential issues described earlier. The datasets used to fit the regression models use these three metrics as input features and use the win % as the output feature.

Since the data in this project is purely numerical, there is not many ethical concerns regarding collection of the data. The main concern is developing the web scraper to be compliant with the basketball-reference robots.txt, which outlines guidelines on what automated programs can and

cannot access from their website. It also specified a crawl delay, to prevent overloading their website with requests. I developed my web scraper to be compliant with these guidelines which is common good practice when developing any type of web scraper. Thankfully, basketball-reference's robots.txt is not very restrictive, allowing me to scrape and parse all the data needed to build my dataset for this project.

Technical Approach

For the regression analysis portion of this project, I decided to use three methods based on in class concepts – Linear Regression, Support Vector Regression, and Random Forest Regression. When evaluating the models, I implemented two different evaluation processes. The first is the standard train/test process. For this, I split the data into a training set which has all team data from the years 2000-2020. The testing set is comprised of data from 2021-2024. The second evaluation process is by evaluating the predicted vs. actual wins for all the teams in a particular season. This will make it easier to show how close the model is getting to the ground truth. For this, I compiled a dataset with all the data (minus the year to evaluate), then made predictions on the remaining year, and compared win totals. For all methods, I use the Root Mean Squared Error (RMSE) and coefficient of determination (R^2) values to evaluate the fit of the regression model. RMSE is just the SSE (defined further below) divided by the number of samples, and then taken the root.

$$MSE = \frac{1}{N} SSE$$

$$RMSE = \sqrt{MSE}$$

The coefficient of determination is a common measure of the fit of a regression model, where 1.0 is a perfect fit, and 0.0 for a constant model which always predicts the mean.

$$R^2 = 1 - \frac{SSE}{TSS}$$

Where,

$$Tot.\ Sum\ of\ Squares\ (TSS) = \sum_{i=1}^{N} (y_i - \mu)^2$$

Linear Regression

The Linear Regression model used is just an implementation on Ordinary Least Squares Regression. The goal of OLS regression is to fit a linear model with some set of coefficients (model parameters) and minimize the sum of squared errors. Thus, the objective function for OLS is as follows

$$SSE = L(\theta) = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$$

Where $\hat{y}_i$ is the predicted value from the model, and y_i is the actual value for the ith sample. Thankfully, the optimal solution for OLS can be solved in closed form, without the use of an iterative algorithm. The optimal solution is as follows

$$w^* = (X^T X)^{-1} X^T y$$

For my implementation, I did not use any transformation functions ϕ nor did I use any type of regularization.

Support Vector Regression

The support vector regression is an extension of the traditional SVM, applied to regression tasks. SVMs are defined by support vectors, which do not care about points that lie beyond the margin. Similarly, the SVR cost function ignores the points that are already close to the model prediction. Training the SVR means solving the following constrained optimization problem

$$minimize \frac{1}{2} ||w||^2$$

$$s.t. |y_i - (w^T x_i + b)| < \varepsilon$$

Where x_i is the training sample, y_i is the target value, and ε is a hyperparameter which serves as a threshold. The idea is that all model predictions $w^T x_i + b$ must fall within the threshold ε from the target value. Just like SVMs in the classification setting, SVRs can take advantage of the kernel trick to efficiently do non-linear regression by applying a kernel function to all pairwise samples. Based on the EDA and preprocessing done with the data, however, there are strong linear relations between by different input features and the response variable. Thus, I focus on the linear kernel. The linear kernel is just the polynomial kernel with a degree of 1, which is defined as follows

$$k(x_i, x_j) = (x_i * x_j)^d = x_i * x_j$$

Out of curiosity, I also run some experiments with gaussian radial basis function (rbf) kernel. The rbf kernel is defined as follows

$$k(x_i, x_j) = \exp\left(\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

For a value of ε, I use 0.1, which seems to be a reasonable threshold.

Random Forest Regression

Random Forest Regression is an ensemble learning method which uses some number of decision trees as weak regressors and uses all their outputs to create a final output. Fundamentally, each decision tree regressor works the same as a decision tree classifier, except that the outputs at the leaf nodes are continuous values rather than discrete class labels.

Just like the decision tree classifiers, the regressor uses a criterion to determine the best split at each tree's node. However, the criterions in the regression setting differ from the classification setting. Common criterion for classifiers are Gini Impurity, or Entropy, which was discussed in class. These are not applicable to the regression setting. The criterion typically used for the regression setting is something like the mean squared error (MSE) or the mean absolute error (MAE). The MSE was defined above, and the MAE can be defined as follows

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |\hat{y}_i - y_i|$$

In my evaluation, I experimented with both MSE and MAE splitting criterions to observe any potential differences in quality of fit for the Random Forest.

Another important hyperparameter for Random Forests are the number of estimators to use. More estimators require higher computation costs but may provide better ensemble performance. Striking a balance between the two is key for using Random Forests. In my experiments, I also explore the effects of different estimator values on the performance.

The last key hyperparameter for the Random Forest is the max depth of the trees. I set this to None which allows the trees to grow until all the leaf nodes are pure. For decision trees, this would typically not be ideal as the regressor would most certainly overfit the training data. However, the Random Forest is an ensemble method which are generally less prone to overfitting than decision trees, since they use many estimators to inform their output, Thus, I did not experiment with the max depth in this project. Had I experienced overfitting problems, this may certainly be something to change but as seen in the Results section, there was no such problems.

Results

The following section outlines the results of the regression models on the data, as well as some comparisons between different hyperparameters. For all experiments done using the "standard" train/test dataset, the train set is comprised of data from 2000-2020, and the test set is comprised of data from 2021-2024.

Linear Regression

For evaluation on the standard train and test sets, I obtain the following results

RMSE = 0.044

R^2 = 0.902

This indicates a good fit from the linear regression model. I also fit the model on data from 2000-2023 and evaluate it on 2024 season data, which is shown in Figure 8.

Evaluation Results for 2024:

	Team	Actual Win%	Predicted Win%	Actual Wins	Predicted Wins
0	Boston Celtics	0.780	0.799662	64.0	65.57
1	Oklahoma City Thunder	0.695	0.693241	57.0	56.85
2	Minnesota Timberwolves	0.683	0.680101	56.0	55.77
3	Denver Nuggets	0.695	0.639107	57.0	52.41
4	New York Knicks	0.610	0.623585	50.0	51.13
5	New Orleans Pelicans	0.598	0.626581	49.0	51.38
6	Los Angeles Clippers	0.622	0.588279	51.0	48.24
7	Philadelphia 76ers	0.573	0.580893	47.0	47.63
8	Phoenix Suns	0.598	0.579077	49.0	47.48
9	Indiana Pacers	0.573	0.563561	47.0	46.21
10	Golden State Warriors	0.561	0.565417	46.0	46.36
11	Milwaukee Bucks	0.598	0.571134	49.0	46.83
12	Cleveland Cavaliers	0.585	0.566194	48.0	46.43
13	Dallas Mavericks	0.610	0.557808	50.0	45.74
14	Orlando Magic	0.573	0.561912	47.0	46.08
15	Miami Heat	0.561	0.553361	46.0	45.38
16	Sacramento Kings	0.561	0.547067	46.0	44.86
17	Houston Rockets	0.500	0.531457	41.0	43.58
18	Los Angeles Lakers	0.573	0.522109	47.0	42.81
19	Chicago Bulls	0.476	0.468441	39.0	38.41
20	Atlanta Hawks	0.439	0.439611	36.0	36.05
21	Brooklyn Nets	0.390	0.424805	32.0	34.83
22	Utah Jazz	0.378	0.366448	31.0	30.05
23	San Antonio Spurs	0.268	0.332494	22.0	27.26
24	Toronto Raptors	0.305	0.327910	25.0	26.89
25	Memphis Grizzlies	0.329	0.313746	27.0	25.73
26	Detroit Pistons	0.171	0.258277	14.0	21.18
27	Washington Wizards	0.183	0.250092	15.0	20.51
28	Portland Trail Blazers	0.256	0.255543	21.0	20.95
29	Charlotte Hornets	0.256	0.220209	21.0	18.06

Figure 8: Predicted and Actual wins via Linear Regression

The Linear Regression model overall does a fairly good job at approximating the win percentages for all teams last season. There are a few exceptions, in cases where teams may have under or over performed during the regular season (see Denver Nuggets or Detroit Pistons).

Support Vector Regression

The evaluation for the SVR experiments are the same as in the linear regression setting. First, I evaluate on my standard train and test sets. The SVR with the linear kernel yields the following results

RMSE = 0.042

R^2 = 0.909

The results are essentially the same if not marginally better than the standard linear regressor using OLS. This is perhaps to be expected since we are using SVR with a standard linear kernel. I also evaluated results on the 2024 season using the same method as before, which is seen in figure 9. Similar to before, the regression model is able to approximate most teams' win percentages quite well, with the exception of a few teams who may have under or over performed.

Evaluation Results for 2024:

	Team	Actual Win%	Predicted Win%	Actual Wins	Predicted Wins
0	Boston Celtics	0.780	0.786167	64.0	64.47
1	Oklahoma City Thunder	0.695	0.687301	57.0	56.36
2	Minnesota Timberwolves	0.683	0.676239	56.0	55.45
3	Denver Nuggets	0.695	0.632486	57.0	51.86
4	New York Knicks	0.610	0.617584	50.0	50.64
5	New Orleans Pelicans	0.598	0.625449	49.0	51.29
6	Los Angeles Clippers	0.622	0.585581	51.0	48.02
7	Philadelphia 76ers	0.573	0.576098	47.0	47.24
8	Phoenix Suns	0.598	0.577327	49.0	47.34
9	Indiana Pacers	0.573	0.556175	47.0	45.61
10	Golden State Warriors	0.561	0.563467	46.0	46.20
11	Milwaukee Bucks	0.598	0.569409	49.0	46.69
12	Cleveland Cavaliers	0.585	0.562430	48.0	46.12
13	Dallas Mavericks	0.610	0.556671	50.0	45.65
14	Orlando Magic	0.573	0.559757	47.0	45.90
15	Miami Heat	0.561	0.550919	46.0	45.18
16	Sacramento Kings	0.561	0.548975	46.0	45.02
17	Houston Rockets	0.500	0.532013	41.0	43.63
18	Los Angeles Lakers	0.573	0.525440	47.0	43.09
19	Chicago Bulls	0.476	0.469519	39.0	38.50
20	Atlanta Hawks	0.439	0.439911	36.0	36.07
21	Brooklyn Nets	0.390	0.426867	32.0	35.00
22	Utah Jazz	0.378	0.371685	31.0	30.48
23	San Antonio Spurs	0.268	0.340035	22.0	27.88
24	Toronto Raptors	0.305	0.332731	25.0	27.28
25	Memphis Grizzlies	0.329	0.321426	27.0	26.36
26	Detroit Pistons	0.171	0.264643	14.0	21.70
27	Washington Wizards	0.183	0.253255	15.0	20.77
28	Portland Trail Blazers	0.256	0.265520	21.0	21.77
29	Charlotte Hornets	0.256	0.228664	21.0	18.75

Figure 9: Predicted and Actual wins via SVR

Just for fun, I also tried using the gaussian radial basis function kernel (rbf) and the polynomial kernel to compare to the linear kernel. Evaluation was done using the standard train and test sets. The results for the rbf kernel are as follows:

RMSE = 0.042

R^2 = 0.908

The results are nearly identical to the linear kernel. Unsurprisingly, introducing a nonlinear kernel doesn't really help the model, since the input features have naturally strong linear relations to the response variable. For the polynomial kernel, I use a degree of 3, as any other values I tried besides 1 are so bad they are not worth mentioning. The results are as follows

RMSE = 0.082

R^2 = 0.652

Clearly increasing the degree of the polynomial kernel beyond 1 is detrimental to the model, mainly for the reasons described above. A standard linear kernel generally does a good job with this dataset.

To demonstrate the effects of the free parameter ε on the results of the SVR, I also use various epsilon values and compare the results. These experiments are done using the linear kernel.

Recall the results of the original baseline $\varepsilon = 0.1$:

RMSE = 0.042

R^2 = 0.909

$\varepsilon = 0.001$:

RMSE = 0.045

R^2 = 0.899

$\varepsilon = 0.01$:

RMSE = 0.044

R^2 = 0.899

$\varepsilon = 0.2$:

RMSE = 0.060

R^2 = 0.815

$\varepsilon = 0.3$:

RMSE = 0.101

R^2 = 0.475

Indeed the value $\varepsilon = 0.1$ has the best performance. One pattern I noticed is that decreasing the epsilon value from 0.1 by orders of magnitude doesn't have much effect on the result but increasing it past 0.1 by relatively small amount has a detrimental impact on the results. Using an epsilon of 0.4 or higher yields an R^2 value of 0.0.

Random Forest Regression

The evaluation of the Random Forest Regression experiments use the same experimental setting as above. In my initial approach, I use the MSE criterion to determine splits and use 100 estimators for the Random Forest. First, I evaluate on my standard train and test sets and obtain the following results.

RMSE = 0.043

R^2 = 0.904

The results for the Random Forest are quite similar to both the SVR and Linear Regression methods. This is not particularly surprising, considering the relation between the input features and the response variable is simple, and has already been shown to be effectively modeled by other simple linear regression models. I also evaluated the results on the 2024 season data using the same method as SVR and Linear Regression. The results are seen in Figure 10. As expected, the model has similar success to the other methods, accurately estimating most teams win percentages with the exception of a few special cases.

Evaluation Results for 2024:

	Team	Actual Win%	Predicted Win%	Actual Wins	Predicted Wins
0	Boston Celtics	0.780	0.77970	64.0	63.94
1	Oklahoma City Thunder	0.695	0.69377	57.0	56.89
2	Minnesota Timberwolves	0.683	0.67694	56.0	55.51
3	Denver Nuggets	0.695	0.63831	57.0	52.34
4	New York Knicks	0.610	0.63603	50.0	52.15
5	New Orleans Pelicans	0.598	0.65980	49.0	54.10
6	Los Angeles Clippers	0.622	0.59681	51.0	48.94
7	Philadelphia 76ers	0.573	0.55789	47.0	45.75
8	Phoenix Suns	0.598	0.60455	49.0	49.57
9	Indiana Pacers	0.573	0.56639	47.0	46.44
10	Golden State Warriors	0.561	0.55801	46.0	45.76
11	Milwaukee Bucks	0.598	0.56234	49.0	46.11
12	Cleveland Cavaliers	0.585	0.56724	48.0	46.51
13	Dallas Mavericks	0.610	0.54812	50.0	44.95
14	Orlando Magic	0.573	0.56323	47.0	46.18
15	Miami Heat	0.561	0.55017	46.0	45.11
16	Sacramento Kings	0.561	0.55699	46.0	45.67
17	Houston Rockets	0.500	0.54019	41.0	44.30
18	Los Angeles Lakers	0.573	0.53020	47.0	43.48
19	Chicago Bulls	0.476	0.44981	39.0	36.88
20	Atlanta Hawks	0.439	0.43400	36.0	35.59
21	Brooklyn Nets	0.390	0.43474	32.0	35.65
22	Utah Jazz	0.378	0.36755	31.0	30.14
23	San Antonio Spurs	0.268	0.31491	22.0	25.82
24	Toronto Raptors	0.305	0.30952	25.0	25.38
25	Memphis Grizzlies	0.329	0.28902	27.0	23.70
26	Detroit Pistons	0.171	0.25351	14.0	20.79
27	Washington Wizards	0.183	0.25505	15.0	20.91
28	Portland Trail Blazers	0.256	0.26557	21.0	21.78
29	Charlotte Hornets	0.256	0.21582	21.0	17.70

Figure 10: Predicted and Actual wins via Random Forest

I also experiment with using the MAE criterion in place of MSE. The results on the train/test set is as follows.

RMSE = 0.044

R^2 = 0.903

The MAE criterion performs marginally worse, but the difference in performance is so small it is hardly notable. I conduct further experiments by varying the number of estimators used in the ensemble regressor. The original results are with 100 estimators. I experiment by increasing and decreasing the number of estimators by one order of magnitude. The results for those experiments on the train/test set is as follows.

n_estimators = 10:

RMSE = 0.045

R^2 = 0.895

n_estimators = 1000:

RMSE = 0.043

R^2 = 0.905

As expected, using just 10 estimators performs the worst on the test set, though it's performance is still very close to the baseline of 100 estimators. Increasing the number of estimators does barely increase performance, but the difference is so small that it is hardly worth the order of magnitude increase in the number trees used.

2025 Predictions

Along with evaluating the regression models on the standard train/test sets and the 2024 season data, I also obtained predictions on the current 2025 season data, so make some very early projections about the final regular season standings. For this experiment, I fit a regression model on data from 2000-2024 and use the fitted model to make predictions for the win% for each NBA team. I use the SVR model with a linear kernel and an epsilon value of 0.1. Table 1 shows the predicted top 5 standings based on the current NBA season data so far.

Team	True W/L%	**Predicted W/L%**	Actual Wins	Predicted Wins	Projected Wins (82 games)
Oklahoma City Thunder	0.773	0.741	17.0	16.31	60.762
Boston Celtics	0.826	0.705	19.0	16.22	57.810
Cleveland Cavaliers	0.870	0.701	20.0	16.12	57.482
Dallas Mavericks	0.652	0.653	15.0	15.03	53.546
New York Knicks	0.636	0.646	14.0	14.21	52.972

Table 1: Predicted Top 5 Standings for the 2025 NBA Season

Interestingly, the team with the best record in the NBA is actually predicted to end up 3rd, and the team with the third best record is predicted to be 1st. The model's predicted win% for the Cavaliers and Celtics is notable lower than their true win percentage. This may indicate that

those teams are actually performing better than their statistical expectations. For the Cavaliers, this is somewhat explainable considering they opened the season with a historic 15 game win streak (T-2nd best all time). As good as the Cavs are as a team, a winning streak like that requires both good play and some extra luck. It remains to be seen how accurate these predictions will end up being. Based on the current play of the teams, this is certainly a reasonable possibility, but the NBA season is long, and many things can happen to alter the course of a team's season.

Discussion

This project put emphasis on collecting and curating your own dataset, which is not something that is often done for ML projects. As mentioned in class, it's common practice to use benchmark datasets to evaluate and compare methodologies easier. These benchmark datasets are often already processed and easily downloadable. For this project, a significant amount of effort and thought was put into how to implement tools to get the data I need and how to transform and preprocess that data to be in a format with can be used for ML tasks. Designing and implementing the data collection and processing tools was a challenging but worthwhile process, as it got me thinking about what features will properly represent the problem I'm trying to tackle, and how best to process and use these features. These are all key steps when working on any machine learning task.

If I had more time, the main improvement I would make would be to add more features. Even though my features used are strong indicators of win%, only three were used. I would also have liked to try and predict playoff success using a combination of team and player data, though this would certainly be a significantly more complicated process. Predicting playoff success is also difficult because there is fewer overall data samples to use. Only about half the teams even make the playoffs, and after each round, half of the teams that played get eliminated. This problem would pose different but interesting problems to try and tackle.

One notable potential weakness of this project is that the three input features used are actually not very independent. Generally, it is desirable to have the input features be good predictors of the response variable and be independent of each other. In practice this isn't always feasible, but in this particular project, there is actually a fairly strong correlation between input features, as illustrated by figure 11. There are regression methods that can address this by introducing various methods of regularization (LASSO or Ridge Regression). These methods are typically used with high dimensional data to avoid multicollinearity issues, so it is unclear if these regularization techniques will actually demonstrate any improvement in my low dimensional dataset. Considering the concept of regularization was also introduced in class, this could certainly be an interesting extension to the project.

References

[1] “Basketball Statistics & History of Every Team & NBA and WNBA Players.” *Basketball Reference*, www.basketball-reference.com/. Accessed 24 Oct. 2024.

[2] “Team Evaluation Metrics - Analytics101.” *NBAstuffer*, 13 July 2023, www.nbastuffer.com/analytics-101/team-evaluation-metrics/.