Predicting Value:

a Regression of NBA Player data on Win Shares

By Will Carnevale

Objectives and Goals

Objective: Build a regression model to predict the win shares contributed by a player based on past performance data

Goal: Learn what aspects of performance are most impactful towards obtaining win shares (and by extrapolation, winning basketball games)

Background: Dean Oliver Hypothesis

1. In 2002, Dean Oliver identified what he considered the most important factors in predicting a basketball team's success:

    Shooting (40%) → positive

    Turnovers (25%) → negative

    Rebounding (20%) → positive

    Free Throws (15%) → positive

2. His model would predict the number of wins a team might get

Background: Win Shares

- My model: **Player-to-player basis** instead of team-basis
- Win share (definition): an estimate of the number of wins contributed by a player
- Total Win Shares = (Offensive Win Shares) + (Defensive win Shares)

Offensive Win Shares equation:

(marginal **offense**) / (marginal points per **win**)

Defensive Win Shares equation:

(marginal **defense**) / (marginal points per **win**)

Tools and Data

Data:

1. NBA data: Obtained data through selenium on the height and weight for each player

2. Basketball Reference: Obtained data for various player stats through pandas scraping methods

Tools:

1. Selenium for primary webscraping

2. Pandas for data manipulation and web-scraping

3. Matplotlib and Seaborn for plotting data

4. Sklearn and Statsmodels.api for regression operations

5. Unidecode to deal with odd characters in strings

First Regression attempt: OLS

```
features = ["TOV","ORB","DRB","AST","STL","BLK","FT_perc","eFG","Height","Weight","MP_pergame"]
X = nba_data[features]
X = sm.add_constant(X)

y = nba_data["WS"]

lm = sm.OLS(y, X)

lm=lm.fit()

lm.summary()
```

	coef	std err	t	P>|t|
const	-17.4714	2.977	-5.868	0.000
TOV	-0.1090	0.026	-4.186	0.000
ORB	0.2090	0.036	5.834	0.000
DRB	0.0356	0.020	1.787	0.075
AST	0.1032	0.012	8.331	0.000
STL	0.2611	0.152	1.713	0.088
BLK	-0.0676	0.060	-1.118	0.264
FT_perc	5.1075	0.964	5.296	0.000
eFG	19.2556	1.474	13.063	0.000
Height	-0.0065	0.040	-0.161	0.872
Weight	0.0079	0.005	1.720	0.086
MP_pergame	0.1191	0.013	8.952	0.000

```
split_and_validate(X,y)
```

```
Validation R^2 score was: 0.6219039706625178
Feature coefficient results:

const : 0.00
TOV : -0.11
ORB : 0.23
DRB : 0.06
AST : 0.10
STL : 0.27
BLK : -0.14
FT_perc : 5.64
eFG : 17.85
Height : -0.02
Weight : 0.01
MP_pergame : 0.12
```

R-squared:	0.731
Adj. R-squared:	0.722

Second Regression attempt: OLS part 2

```
features = ["TOV","ORB","DRB","AST","FT_perc","eFG","MP_pergame"]
X = nba_data[features]
X = sm.add_constant(X)

y = nba_data["WS"]

lm = sm.OLS(y, X)

lm=lm.fit()

lm.summary()
```

	coef	std err	t	P>|t|
const	-16.2722	1.073	-15.159	0.000
TOV	-0.1020	0.026	-3.949	0.000
ORB	0.2170	0.033	6.567	0.000
DRB	0.0404	0.017	2.387	0.018
AST	0.1056	0.011	9.348	0.000
FT_perc	5.3483	0.959	5.576	0.000
eFG	18.9049	1.454	13.001	0.000
MP_pergame	0.1207	0.013	9.194	0.000

```
Validation R^2 score was: 0.6624951493184974
Feature coefficient results:

const : 0.00
TOV : -0.11
ORB : 0.22
DRB : 0.06
AST : 0.10
FT_perc : 6.02
eFG : 17.23
MP_pergame : 0.12
```

R-squared:	0.725
Adj. R-squared:	0.719

Regression attempt 2.5: Feature Engineering

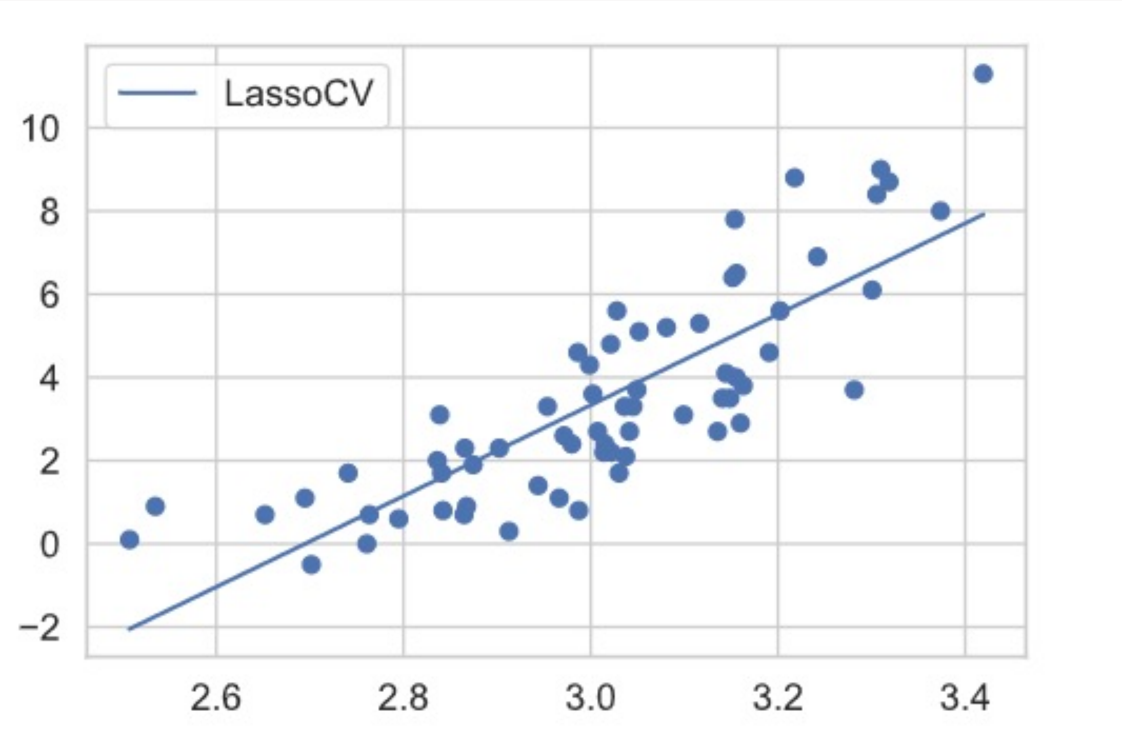
```
pd.get_dummies(nba_data['Pos']).head()
```

	C	PF	PG	SF	SG
0	0	1	0	0	0
2	1	0	0	0	0
3	1	0	0	0	0
4	1	0	0	0	0
6	0	0	0	0	1

```
X2 = X.copy()

X2['Pos'] = nba_data['Pos']

split_and_validate(pd.get_dummies(X2, drop_first=True), y)
```

```
Validation R^2 score was: 0.6249014975812022
Feature coefficient results:

TOV : -0.11
ORB : 0.25
DRB : 0.04
AST : 0.12
FT_perc : 6.57
eFG : 17.64
MP_pergame : 0.11
Pos_PF : 0.43
Pos_PG : -0.34
Pos_SF : 0.75
Pos_SG : -0.11
```

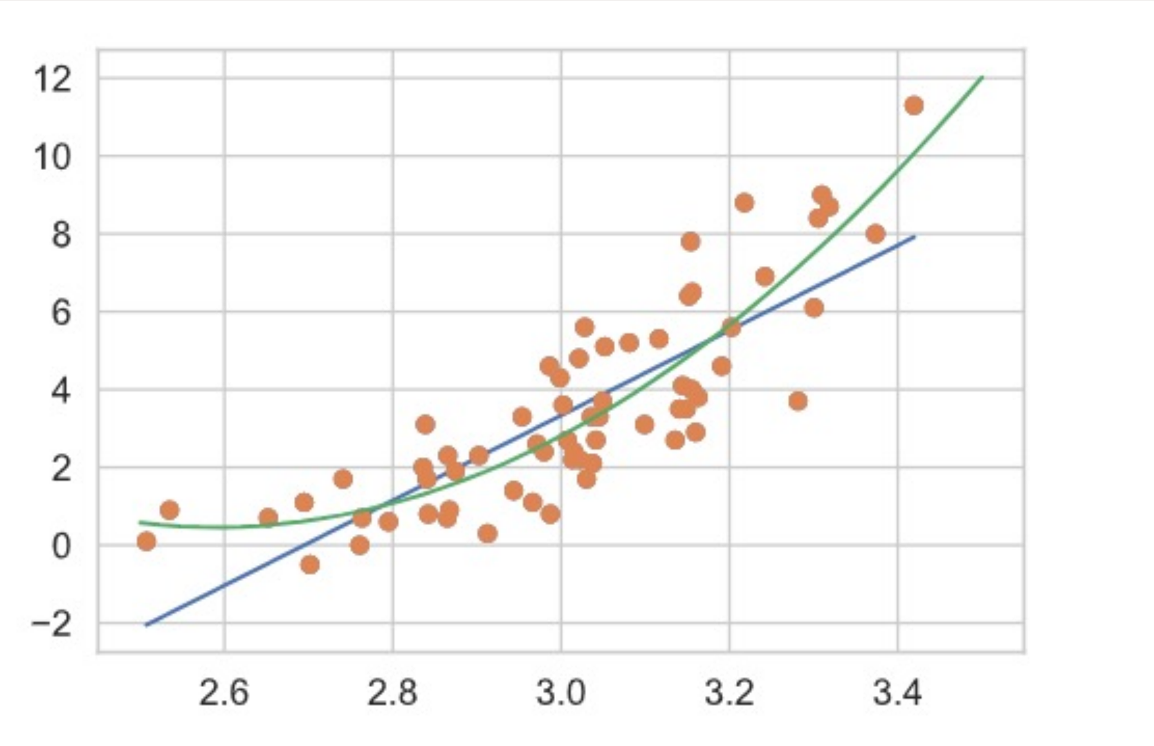
Third Regression attempt: LassoCV




```
: list(zip(selected_columns, lasso_model.coef_))

: [('TOV', -0.0),
   ('ORB', 0.0),
   ('DRB', 0.0),
   ('AST', 0.0),
   ('FT_perc', 0.0),
   ('eFG', 0.13342401856902697),
   ('MP_pergame', 0.11592546602047331)]
```

```
]: 1-r2_score((m*x) + b,x)

]: 0.8774069303427643
```

Fourth Regression attempt: Polynomial




```
1-r2_score(poly(t),t)
```

```
0.9266067974093367
```

Conclusions + Future Work:

```
: list(zip(selected_columns, lasso_model.coef_))

: [('TOV', -0.0),
   ('ORB', 0.0),
   ('DRB', 0.0),
   ('AST', 0.0),
   ('FT_perc', 0.0),
   ('eFG', 0.13342401856902697),
   ('MP_pergame', 0.11592546602047331)]
```

Conclusions:

1. According to the LassoCV model, Effective Field Goal % has the highest regression coefficient (Thus it is the the best predictor of win shares)
2. The next highest coefficient is minutes
3. Rest of features were zeroed out by LassoCV

    Interpretation: "Offense is more important than defense"

Future work:

1. Mirroring the weights given by Dean Oliver in my feature engineering and applying it to the appropriate data

Afterthought:

Did a 5-fold cross validation between regression attempts 2.5 and 3 to make sure my model generalized well

```
#cross validation ---> check to make sure generalization is ok
from sklearn.model_selection import cross_val_score
lm = LinearRegression()

cross_val_score(lm, X, y, # estimator, features, target
                cv=5, # number of folds
                scoring='r2') # scoring metric
```

```
array([0.69371621, 0.65476151, 0.72956563, 0.73524183, 0.68581334])
```