Predicting Baseball Season Wins.

The objective of this notebook is to provide a overview of what componenets create a winning team and make recommendations based on linear regression models to assist in building a competitive MLB Team.

```
# Importing relevant libraries and dataframes
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline


from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
import statsmodels.api as sm
from sklearn.metrics import mean_squared_error
from scipy.stats import linregress
import math


#BaseballRef
https://www.baseball-reference.com/leagues/majors/2022.shtml
br_2016_batting = pd.read_csv('data/br_2016_batting.csv')
br_2016_fielding = pd.read_csv('data/br_2016_fielding.csv')
br_2016_pitching = pd.read_csv('data/br_2016_pitching.csv')
br_2016_standings = pd.read_csv('data/br_2016_standings.csv')
br_2016_war = pd.read_csv('data/br_2016_wins_replacment.csv')
br_2017_batting = pd.read_csv('data/br_2017_batting.csv')
br_2017_fielding = pd.read_csv('data/br_2017_fielding.csv')
br_2017_pitching = pd.read_csv('data/br_2017_pitching.csv')
br_2017_standings = pd.read_csv('data/br_2017_standings.csv')
br_2017_war = pd.read_csv('data/br_2017_wins_replacment.csv')
br_2018_batting = pd.read_csv('data/br_2018_batting.csv')
br_2018_fielding = pd.read_csv('data/br_2018_fielding.csv')
br_2018_pitching = pd.read_csv('data/br_2018_pitching.csv')
br_2018_standings = pd.read_csv('data/br_2018_standings.csv')
br_2018_war = pd.read_csv('data/br_2018_wins_replacment.csv')
br_2019_batting = pd.read_csv('data/br_2019_batting.csv')
br_2019_fielding = pd.read_csv('data/br_2019_fielding.csv')
br_2019_pitching = pd.read_csv('data/br_2019_pitching.csv')
br_2019_standings = pd.read_csv('data/br_2019_standings.csv')
br_2019_war = pd.read_csv('data/br_2019_wins_replacment.csv')
br_2020_batting = pd.read_csv('data/br_2020_batting.csv')
br_2020_fielding = pd.read_csv('data/br_2020_fielding.csv')
br_2020_pitching = pd.read_csv('data/br_2020_pitching.csv')
```

```
br_2020_standings = pd.read_csv('data/br_2020_standings.csv')
br_2020_war = pd.read_csv('data/br_2020_wins_replacment.csv')
br_2021_batting = pd.read_csv('data/br_2021_batting.csv')
br_2021_fielding = pd.read_csv('data/br_2021_fielding.csv')
br_2021_pitching = pd.read_csv('data/br_2021_pitching.csv')
br_2021_standings = pd.read_csv('data/br_2021_standings.csv')
br_2021_war = pd.read_csv('data/br_2021_wins_replacment.csv')
br_2022_batting = pd.read_csv('data/br_2022_batting.csv')
br_2022_fielding = pd.read_csv('data/br_2022_fielding.csv')
br_2022_pitching = pd.read_csv('data/br_2022_pitching.csv')
br_2022_standings = pd.read_csv('data/br_2022_standings.csv')
br_2022_war = pd.read_csv('data/br_2022_wins_replacment.csv')

#Baseball Cube
bc_2022_salary = pd.read_csv('data/baseball_cube_2022.csv')
bc_2021_salary = pd.read_csv('data/baseball_cube_2021.csv')
bc_2020_salary = pd.read_csv('data/baseball_cube_2020.csv')
bc_2019_salary = pd.read_csv('data/baseball_cube_2019.csv')
bc_2018_salary = pd.read_csv('data/baseball_cube_2018.csv')
bc_2017_salary = pd.read_csv('data/baseball_cube_2017.csv')
bc_2016_salary = pd.read_csv('data/baseball_cube_2016.csv')


pd.set_option('max_columns', None)
```

Data and Business Understanding

- This data set is using two different sources for our models
- Baseball Reference is where we pulled our seasonal statisics from https://www.baseball-reference.com/leagues/majors/2022.shtml
- Baseball Cube is where we pulled our record and payroll from https://www.thebaseballcube.com/
- The data goes over batting, pitching, and fielding statistics from 2016 to 2022
- It covers all 30 MLB teams

Limitations

- Covid Season
- No minor leagues or international leauges

Main DF creation

Below we are merging, dropping, and editing our csv's into one dataframe to model from.

```
#Creating Tm column to merge with baseball reference dfs
bc_2022_salary['Tm'] = bc_2022_salary['team name']
```

```
bc_2021_salary['Tm'] = bc_2021_salary['team name']
bc_2020_salary['Tm'] = bc_2020_salary['team name']
bc_2019_salary['Tm'] = bc_2019_salary['team name']
bc_2018_salary['Tm'] = bc_2018_salary['team name']
bc_2017_salary['Tm'] = bc_2017_salary['team name']
bc_2016_salary['Tm'] = bc_2016_salary['team name']

#creating year column for salary graphs
bc_2022_salary['year'] = 2022
bc_2021_salary['year'] = 2021
bc_2020_salary['year'] = 2020
bc_2019_salary['year'] = 2019
bc_2018_salary['year'] = 2018
bc_2017_salary['year'] = 2017
bc_2016_salary['year'] = 2016

#Merging dfs
df_2022 = br_2022_standings.merge(br_2022_batting, on
='Tm').merge(br_2022_pitching, on ='Tm').merge(br_2022_fielding, on
='Tm').merge(bc_2022_salary, on = 'Tm')
df_2021 = br_2021_standings.merge(br_2021_batting, on
='Tm').merge(br_2021_pitching, on ='Tm').merge(br_2021_fielding, on
='Tm').merge(bc_2021_salary, on = 'Tm')
df_2020 = br_2020_standings.merge(br_2020_batting, on
='Tm').merge(br_2020_pitching, on ='Tm').merge(br_2020_fielding, on
='Tm').merge(bc_2020_salary, on = 'Tm')
df_2019 = br_2019_standings.merge(br_2019_batting, on
='Tm').merge(br_2019_pitching, on ='Tm').merge(br_2019_fielding, on
='Tm').merge(bc_2019_salary, on = 'Tm')
df_2018 = br_2018_standings.merge(br_2018_batting, on
='Tm').merge(br_2018_pitching, on ='Tm').merge(br_2018_fielding, on
='Tm').merge(bc_2018_salary, on = 'Tm')
df_2017 = br_2017_standings.merge(br_2017_batting, on
='Tm').merge(br_2017_pitching, on ='Tm').merge(br_2017_fielding, on
='Tm').merge(bc_2017_salary, on = 'Tm')
df_2016 = br_2016_standings.merge(br_2016_batting, on
='Tm').merge(br_2016_pitching, on ='Tm').merge(br_2016_fielding, on
='Tm').merge(bc_2016_salary, on = 'Tm')

#concating dfs
df = pd.concat([df_2016, df_2017, df_2018, df_2019, df_2020, df_2021,
df_2022], ignore_index=True)
df_salary = pd.concat([bc_2022_salary, bc_2021_salary, bc_2019_salary,
bc_2018_salary, bc_2017_salary, bc_2016_salary], ignore_index=True)

#striping spaces, commas, and setting payroll as float
df.columns = df.columns.str.replace(' ', '')
df["teampayroll"] = df["teampayroll"].str.replace(",","").astype(int)
df["lastyrpayroll"] =
df["lastyrpayroll"].str.replace(",","").astype(int)
```

```
df["w"] = df["w"].astype(int)

df_salary.columns = df_salary.columns.str.replace(' ', '')
df_salary["teampayroll"] =
df_salary["teampayroll"].str.replace(",","").astype(float)
df_salary["lastyrpayroll"] =
df_salary["lastyrpayroll"].str.replace(",","").astype(float)
df_salary["w"] = df_salary["w"].astype(float)

#dropping columns that directly correlate to Ws/Winpercentage
columns_drop = df[['W_x', 'W-L%_x', 'Rk', 'L_x', 'pythWL', 'Luck',
'vEast', 'vCent', 'vWest', 'Inter', 'Home', 'Road', 'ExInn'
                  , '1Run', 'vRHP', 'vLHP', '≥.500', '<.500', 'W_y',
'L_y', 'W-L%_y', 'Strk', 'last10', 'last20', 'last30',
                  'roster', 'league', 'division', 'rank', 'lgrk',
'mlbrk', 'topsalary', 'teamname', 'l']]
df.drop(columns=columns_drop, inplace=True)

#Creating a money spent per win statistic
df["perwin"] = df["teampayroll"].div(df["w"])
df_salary["perwin"] = df_salary["teampayroll"].div(df_salary["w"])

#setting index to Team name
df = df.set_index('Tm')
df_salary = df_salary.set_index('teamname')

#checking correlations to wins
corr = df.corr()['w']
corr.abs().sort_values(ascending=False)[1:].head(10)
```

```
R_y      0.913771
RBI      0.913027
SO_y     0.889851
SV       0.885519
BB_x     0.881786
TB       0.875384
LOB_x    0.843682
H_x      0.843076
PA       0.835271
2B       0.827519
Name: w, dtype: float64
```

```
#checking correlations to win percentage
corr = df.corr()['wpct']
corr.abs().sort_values(ascending=False)[1:].head(10)
```

```
Rdiff     0.948654
SRS       0.927847
ERA+      0.822787
ERA       0.804416
RA/G_x    0.803054
```

```
RA/G_y    0.803054
RA        0.797957
WHIP      0.772331
FIP       0.723026
H9        0.701982
Name: wpct, dtype: float64
```

```
#checking correlations to payroll
corr = df.corr()['teampayroll']
corr.abs().sort_values(ascending=False)[1:].head(10)
```

```
lastyrpayroll    0.817399
PAge             0.485671
BatAge           0.465600
perwin           0.456952
SRS              0.438216
Rdiff            0.432428
OBP              0.409236
R_x              0.402791
R/G              0.399577
wpct             0.394345
Name: teampayroll, dtype: float64
```

```
#checking correlations to perwin
corr = df.corr()['perwin']
corr.abs().sort_values(ascending=False)[1:].head(15)
```

```
GF       0.757010
Ch       0.755661
PO       0.755179
IP       0.755147
Inn      0.755133
GS_y     0.754553
G_y      0.754553
G_x      0.754553
GS_x     0.754553
G        0.754553
BF       0.748452
AB       0.748135
PA       0.747764
SO_x     0.740127
LOB_y    0.734635
Name: perwin, dtype: float64
```

Baseline Model

This model was is using all of teh columns to predicts wins, this is mainly to see what pvalues are sub .05 and to select those columns for our next model

```
#creating first simple model to check pvalues
model = df.copy()

X = model.drop('w', axis=1)
y = model['w']

y = y.values.reshape(-1,1)



X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)

win_model = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = win_model.fit()
results

results.rsquared

print(results.summary())

#R_x, R/G, R_y, H_x, BA, OBP, TB, LOB_x, RA/G_x, G_y, GS_x, R, BB_y,
SO_y, BF, WHIP, SO9, LOB_y, RA/G_y, G, GS_y, Fld%, Rtot/yr
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:                      y   R-squared:
0.999
Model:                            OLS   Adj. R-squared:
0.999
Method:                 Least Squares   F-statistic:
1514.
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
8.77e-114
Time:                        16:01:37   Log-Likelihood:
-149.05
No. Observations:                 168   AIC:
460.1
Df Residuals:                      87   BIC:
713.1
Df Model:                          80

Covariance Type:            nonrobust

======================================================================
===========
```

	coef	std err	t	P>|t|	[0.025	0.975]
const	336.8109	264.087	1.275	0.206	-188.091	861.713
R_x	5.4960	2.717	2.022	0.046	0.095	10.897
RA	-2.6913	3.293	-0.817	0.416	-9.236	3.854
Rdiff	4.5772	3.177	1.441	0.153	-1.738	10.892
SOS	-0.1368	2.126	-0.064	0.949	-4.363	4.090
SRS	0.5282	2.160	0.245	0.807	-3.765	4.822
#Bat	0.0085	0.161	0.053	0.958	-0.312	0.329
BatAge	0.0551	0.105	0.527	0.599	-0.153	0.263
R/G	-18.3782	5.237	-3.509	0.001	-28.787	-7.969
G_x	0.0058	0.002	3.138	0.002	0.002	0.009
PA	-0.0275	0.047	-0.587	0.559	-0.121	0.066
AB	-0.0494	0.043	-1.139	0.258	-0.136	0.037
R_y	0.1307	0.026	4.943	0.000	0.078	0.183
H_x	-0.0323	0.015	-2.195	0.031	-0.062	-0.003
2B	-3.26e-05	0.005	-0.006	0.995	-0.010	0.010
3B	0.0177	0.011	1.671	0.098	-0.003	0.039
HR_x	0.0069	0.007	0.964	0.338	-0.007	0.021
RBI	-0.0089	0.016	-0.563	0.575	-0.040	0.022
SB	-0.0016	0.005	-0.337	0.737	-0.011	0.008
CS	-0.0067	0.019	-0.346	0.730	-0.045	0.032
BB_x	-0.0153	0.043	-0.355	0.724	-0.101	0.071
SO_x	-0.0013	0.001	-1.012	0.314	-0.004	0.001
BA	415.1196	84.145	4.933	0.000	247.872	582.367

OBP	-354.2272	182.392	-1.942	0.055	-716.751	8.297
SLG	-225.9859	183.730	-1.230	0.222	-591.168	139.197
OPS	98.8637	180.284	0.548	0.585	-259.470	457.197
OPS+	-0.0284	0.048	-0.597	0.552	-0.123	0.066
TB	0.0238	0.010	2.381	0.019	0.004	0.044
GDP	-0.0003	0.011	-0.031	0.975	-0.023	0.022
HBP_x	-0.0072	0.043	-0.167	0.868	-0.093	0.079
SH	-0.0444	0.043	-1.037	0.303	-0.129	0.041
SF	-0.0578	0.048	-1.211	0.229	-0.153	0.037
IBB_x	0.0076	0.012	0.616	0.539	-0.017	0.032
LOB_x	0.0697	0.020	3.405	0.001	0.029	0.110
#P	-0.0448	0.039	-1.150	0.253	-0.122	0.033
PAge	-0.0049	0.100	-0.049	0.961	-0.204	0.194
RA/G_x	9.0779	3.347	2.712	0.008	2.425	15.731
ERA	3.7262	3.797	0.981	0.329	-3.822	11.274
G_y	0.0058	0.002	3.138	0.002	0.002	0.009
GS_x	0.0058	0.002	3.138	0.002	0.002	0.009
GF	0.0172	0.037	0.465	0.643	-0.056	0.090
CG_x	-0.0114	0.037	-0.305	0.761	-0.086	0.063
tSho	0.0339	0.034	0.992	0.324	-0.034	0.102
cSho	0.0822	0.138	0.597	0.552	-0.191	0.356
SV	0.0022	0.023	0.096	0.924	-0.043	0.047
IP	0.4078	0.427	0.955	0.342	-0.441	1.257
H_y	0.0182	0.012	1.555	0.124	-0.005	0.041
R	-0.1318	0.037	-3.556	0.001	-0.206	-0.058

ER	-0.0178	0.029	-0.612	0.542	-0.075	0.040
HR_y	-0.0379	0.031	-1.226	0.223	-0.099	0.024
BB_y	0.0301	0.014	2.114	0.037	0.002	0.058
IBB_y	-0.0129	0.011	-1.205	0.231	-0.034	0.008
SO_y	-0.0183	0.007	-2.563	0.012	-0.032	-0.004
HBP_y	0.0068	0.016	0.420	0.675	-0.025	0.039
BK	0.0264	0.035	0.746	0.458	-0.044	0.097
WP	-0.0065	0.008	-0.813	0.419	-0.022	0.009
BF	0.0628	0.031	2.019	0.047	0.001	0.125
ERA+	-0.0036	0.035	-0.102	0.919	-0.074	0.067
FIP	0.9058	3.696	0.245	0.807	-6.440	8.252
WHIP	-50.9156	27.151	-1.875	0.064	-104.880	3.049
H9	1.4671	2.713	0.541	0.590	-3.926	6.860
HR9	4.3967	3.135	1.403	0.164	-1.834	10.627
BB9	-1.6588	2.682	-0.619	0.538	-6.989	3.671
SO9	3.1333	1.247	2.512	0.014	0.654	5.613
SO/W	-2.4927	1.758	-1.418	0.160	-5.987	1.002
LOB_y	-0.0628	0.030	-2.111	0.038	-0.122	-0.004
#Fld	0.0128	0.163	0.078	0.938	-0.312	0.338
RA/G_y	9.0779	3.347	2.712	0.008	2.425	15.731
DefEff	-36.0920	83.663	-0.431	0.667	-202.381	130.197
G	0.0058	0.002	3.138	0.002	0.002	0.009
GS_y	0.0519	0.017	3.138	0.002	0.019	0.085
CG_y	0.0011	0.002	0.646	0.520	-0.002	0.005
Inn	-0.0134	0.080	-0.167	0.867	-0.172	0.145

```
Ch                 -0.0310      0.043     -0.722      0.472      -0.116
0.054
PO                 -0.0505      0.126     -0.401      0.690      -0.301
0.200
A                   0.0294      0.043      0.679      0.499      -0.057
0.115
E                  -0.0099      0.044     -0.226      0.822      -0.098
0.078
DP                 -0.0006      0.010     -0.058      0.954      -0.020
0.019
Fld%             -231.9965    125.034     -1.855      0.067    -480.516
16.523
Rtot                0.0222      0.016      1.377      0.172      -0.010
0.054
Rtot/yr            -0.2862      0.162     -1.765      0.081      -0.608
0.036
Rdrs                0.0006      0.004      0.152      0.879      -0.007
0.008
Rdrs/yr            -0.0408      0.062     -0.657      0.513      -0.164
0.083
Rgood               0.0200      0.021      0.967      0.336      -0.021
0.061
teampayroll   -2.935e-09   5.58e-09     -0.526      0.600    -1.4e-08
8.15e-09
wpct              123.7699      7.387     16.755      0.000     109.087
138.453
lastyrpayroll -2.274e-10   3.57e-09     -0.064      0.949   -7.32e-09
6.86e-09
year               -0.0534      0.110     -0.485      0.629      -0.272
0.166
perwin          8.243e-08   2.43e-07      0.339      0.736   -4.01e-07
5.66e-07
======================================================================
========
Omnibus:                       22.680   Durbin-Watson:
2.192
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
43.590
Skew:                          -0.637   Prob(JB):
3.42e-10
Kurtosis:                       5.146   Cond. No.
1.06e+16
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The smallest eigenvalue is 5.83e-14. This might indicate that
there are
```

```
strong multicollinearity problems or that the design matrix is
singular.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)



MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)


print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Mean Square Error:

4.40907866806475

Root Mean Square Error:

2.0997806237949597
```

Second Win Model w/ pvalues

This model pulls all the significant pvalues from the previous model and puts those columns in our X, this performs almost exactly the same as the above model with only a difference of ..034 in our Adj R-squared value.

```
model = df.copy()

X = model[['R_x', 'R/G', 'R_y', 'H_x', 'BA', 'OBP', 'TB', 'LOB_x',
'RA/G_x', 'G_y', 'GS_x', 'R', 'BB_y', 'SO_y'
           , 'BF', 'WHIP', 'SO9', 'LOB_y', 'RA/G_y', 'G', 'GS_y', 'Fld
%', 'Rtot/yr']]
y = model['w']


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)
```

```
win_model2 = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = win_model2.fit()
results

results.rsquared

print(results.summary())
                            OLS Regression Results

======================================================================
========
Dep. Variable:                      w   R-squared:
0.969
Model:                            OLS   Adj. R-squared:
0.965
Method:                 Least Squares   F-statistic:
246.0
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
9.29e-102
Time:                        16:01:37   Log-Likelihood:
-464.55
No. Observations:                 168   AIC:
969.1
Df Residuals:                     148   BIC:
1032.
Df Model:                          19

Covariance Type:            nonrobust

======================================================================
========
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
--------
const        -25.3555    142.642     -0.178      0.859    -307.234
256.523
R_x            7.6004     11.357      0.669      0.504     -14.843
30.044
R/G           -9.4315     11.981     -0.787      0.432     -33.107
14.244
R_y            0.1093      0.032      3.441      0.001       0.047
0.172
H_x           -0.0159      0.023     -0.681      0.497      -0.062
0.030
BA            79.5868    149.303      0.533      0.595    -215.453
374.627
OBP            7.4565     91.925      0.081      0.935    -174.199
```

```
189.112
TB            -0.0030      0.007     -0.435      0.664      -0.017
0.011
LOB_x         -0.0068      0.013     -0.516      0.607      -0.033
0.019
RA/G_x        -2.7632      1.899     -1.455      0.148      -6.515
0.989
G_y           -0.0006      0.003     -0.183      0.855      -0.007
0.006
GS_x          -0.0006      0.003     -0.183      0.855      -0.007
0.006
R             -0.0902      0.020     -4.487      0.000      -0.130
-0.050
BB_y           0.0097      0.013      0.771      0.442      -0.015
0.035
SO_y           0.0148      0.016      0.911      0.364      -0.017
0.047
BF             0.0228      0.011      2.115      0.036       0.002
0.044
WHIP          12.3106     15.672      0.785      0.433     -18.660
43.281
SO9           -2.3003      2.467     -0.932      0.353      -7.176
2.575
LOB_y         -0.0524      0.021     -2.452      0.015      -0.095
-0.010
RA/G_y        -2.7632      1.899     -1.455      0.148      -6.515
0.989
G             -0.0006      0.003     -0.183      0.855      -0.007
0.006
GS_y          -0.0054      0.029     -0.183      0.855      -0.064
0.053
Fld%          43.7689    137.062      0.319      0.750    -227.082
314.620
Rtot/yr       -0.1654      0.162     -1.023      0.308      -0.485
0.154
======================================================================
========
Omnibus:                        7.549   Durbin-Watson:
2.272
Prob(Omnibus):                  0.023   Jarque-Bera (JB):
8.928
Skew:                           0.327   Prob(JB):
0.0115
Kurtosis:                       3.921   Cond. No.
2.46e+20
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
```

```
correctly specified.
[2] The smallest eigenvalue is 1.28e-31. This might indicate that
there are
strong multicollinearity problems or that the design matrix is
singular.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)



MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)



print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Mean Square Error:

10.697219343378546

Root Mean Square Error:

3.270660383374976
```

Third Win Model w/ p values

```
model = df.copy()

X = model[['R_x',  'H_x', 'BA', 'OBP', 'TB', 'LOB_x', 'RA/G_x', 'G_y',
'GS_x', 'BB_y', 'SO_y'
           , 'BF', 'WHIP', 'SO9', 'LOB_y', 'Fld%']]
y = model['W']


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)
```

```
win_model2 = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = win_model2.fit()
results

results.rsquared

print(results.summary())
                            OLS Regression Results

==============================================================================
========
Dep. Variable:                      w   R-squared:
0.961
Model:                            OLS   Adj. R-squared:
0.957
Method:                 Least Squares   F-statistic:
248.4
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
1.14e-98
Time:                        16:01:37   Log-Likelihood:
-485.08
No. Observations:                 168   AIC:
1002.
Df Residuals:                     152   BIC:
1052.
Df Model:                          15

Covariance Type:            nonrobust

==============================================================================
========
                 coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
--------
const        128.6852    151.754      0.848      0.398    -171.134
428.504
R_x           10.4788      2.465      4.251      0.000       5.609
15.348
H_x            0.0294      0.020      1.492      0.138      -0.010
0.068
BA          -227.7152    116.019     -1.963      0.052    -456.934
1.503
OBP           56.0784     96.089      0.584      0.560    -133.765
245.922
TB             0.0066      0.007      0.930      0.354      -0.007
0.021
LOB_x         -0.0009      0.014     -0.067      0.947      -0.028
```

```
0.026
RA/G_x        -16.4423      2.630     -6.251      0.000     -21.639
-11.245
G_y             0.2167      0.125      1.731      0.086      -0.031
0.464
GS_x            0.2167      0.125      1.731      0.086      -0.031
0.464
BB_y           -0.0018      0.012     -0.157      0.876      -0.025
0.021
SO_y            0.0607      0.015      3.929      0.000       0.030
0.091
BF             -0.0160      0.010     -1.662      0.099      -0.035
0.003
WHIP           45.2079     16.501      2.740      0.007      12.606
77.810
SO9            -7.7148      2.360     -3.269      0.001     -12.378
-3.052
LOB_y          -0.0264      0.021     -1.267      0.207      -0.068
0.015
Fld%          -55.4308    148.942     -0.372      0.710    -349.695
238.833
======================================================================
========
Omnibus:                        5.931   Durbin-Watson:
2.232
Prob(Omnibus):                  0.052   Jarque-Bera (JB):
7.854
Skew:                           0.186   Prob(JB):
0.0197
Kurtosis:                       3.992   Cond. No.
1.31e+19
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The smallest eigenvalue is 4.27e-29. This might indicate that
there are
strong multicollinearity problems or that the design matrix is
singular.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)
```

```
MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)



print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Mean Square Error:

15.71482795723181

Root Mean Square Error:

3.9641932290482274
```

Fourth Win Model w/highest correlation

```
model = df.copy()

X = model[['R_y', 'RBI', 'SO_y', 'SV', 'BB_x', 'TB', 'LOB_x', 'H_x',
'PA']]
y = model['w']


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)

win_model2 = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = win_model2.fit()
results

results.rsquared

print(results.summary())
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:                      w   R-squared:
0.945
Model:                            OLS   Adj. R-squared:
```

```
0.942
Method:                 Least Squares   F-statistic:
302.3
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
8.94e-95
Time:                        16:01:37   Log-Likelihood:
-513.37
No. Observations:                 168   AIC:
1047.
Df Residuals:                     158   BIC:
1078.
Df Model:                           9

Covariance Type:            nonrobust

======================================================================
========
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
--------
const         -1.4420      1.751     -0.823      0.411      -4.900
2.016
R_y            0.1092      0.071      1.545      0.124      -0.030
0.249
RBI            0.0321      0.072      0.446      0.656      -0.110
0.175
SO_y           0.0219      0.004      5.023      0.000       0.013
0.030
SV             0.8416      0.063     13.441      0.000       0.718
0.965
BB_x           0.0140      0.017      0.825      0.411      -0.019
0.047
TB            -0.0337      0.009     -3.872      0.000      -0.051
-0.016
LOB_x          0.0082      0.018      0.445      0.657      -0.028
0.045
H_x            0.0223      0.017      1.342      0.181      -0.010
0.055
PA            -0.0089      0.003     -3.035      0.003      -0.015
-0.003
======================================================================
========
Omnibus:                        0.773   Durbin-Watson:
1.982
Prob(Omnibus):                  0.679   Jarque-Bera (JB):
0.499
Skew:                           0.116   Prob(JB):
0.779
Kurtosis:                       3.132   Cond. No.
```

```
2.81e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.81e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)

y_pred = reg.predict(X_test)



MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)



print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Mean Square Error:

27.13850968748008

Root Mean Square Error:

5.209463474051822
```

Base Model for Win percentage

Here I am just changing the target from wins to win percentage and seeing what colums to pull out that have pvalues sub .05

```
model = df.copy()

X = model.drop('wpct', axis=1)
```

```
y = model['wpct']

y = y.values.reshape(-1,1)


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)

wpct_model = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = wpct_model.fit()
results

results.rsquared

print(results.summary())

#R_x, R/G, G_x, H_x, HR_x, BA, TB, RA/G_x, G_y, GS_x, BB_y, SO_y,
WHIP, SO9, RA/G_y, G, GS_y, Fld%, Rtot/yr
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:                      y   R-squared:
0.998
Model:                            OLS   Adj. R-squared:
0.996
Method:                 Least Squares   F-statistic:
476.5
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
5.42e-92
Time:                        16:01:37   Log-Likelihood:
683.12
No. Observations:                 168   AIC:
-1204.
Df Residuals:                      87   BIC:
-951.2
Df Model:                          80

Covariance Type:            nonrobust

======================================================================
===========
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
-----------
const         -2.7494      1.858     -1.479      0.143      -6.443
0.944
```

R_x	-0.0471	0.019	-2.481	0.015	-0.085	-0.009
RA	-0.0164	0.023	-0.703	0.484	-0.063	0.030
Rdiff	-0.0238	0.023	-1.057	0.293	-0.069	0.021
SOS	0.0004	0.015	0.024	0.981	-0.029	0.030
SRS	-0.0030	0.015	-0.199	0.843	-0.033	0.027
#Bat	-0.0005	0.001	-0.420	0.675	-0.003	0.002
BatAge	-0.0003	0.001	-0.434	0.666	-0.002	0.001
R/G	0.1250	0.037	3.365	0.001	0.051	0.199
G_x	-4.109e-05	1.3e-05	-3.170	0.002	-6.69e-05	-1.53e-05
PA	-0.0003	0.000	-0.766	0.446	-0.001	0.000
AB	0.0003	0.000	0.873	0.385	-0.000	0.001
R_y	-0.0004	0.000	-1.724	0.088	-0.001	5.48e-05
H_x	0.0003	0.000	3.359	0.001	0.000	0.001
2B	-2.669e-05	3.63e-05	-0.736	0.464	-9.87e-05	4.54e-05
3B	-0.0001	7.5e-05	-1.659	0.101	-0.000	2.46e-05
HR_x	-0.0001	4.94e-05	-2.191	0.031	-0.000	-1e-05
RBI	0.0001	0.000	0.949	0.345	-0.000	0.000
SB	2.69e-05	3.37e-05	0.799	0.427	-4e-05	9.38e-05
CS	-0.0001	0.000	-0.757	0.451	-0.000	0.000
BB_x	-3.702e-05	0.000	-0.121	0.904	-0.001	0.001
SO_x	9.312e-06	9.37e-06	0.994	0.323	-9.31e-06	2.79e-05
BA	-3.9527	0.522	-7.579	0.000	-4.989	-2.916
OBP	2.1863	1.294	1.689	0.095	-0.386	4.759
SLG	1.4677	1.299	1.130	0.262	-1.114	4.049
OPS	0.0054	1.275	0.004	0.997	-2.529	2.539

OPS+	0.0002	0.000	0.484	0.630	-0.001	0.001
TB	-0.0003	6.72e-05	-3.911	0.000	-0.000	-0.000
GDP	4.366e-06	7.88e-05	0.055	0.956	-0.000	0.000
HBP_x	-7.534e-05	0.000	-0.247	0.806	-0.001	0.001
SH	0.0002	0.000	0.678	0.499	-0.000	0.001
SF	0.0003	0.000	1.034	0.304	-0.000	0.001
IBB_x	1.156e-05	8.73e-05	0.132	0.895	-0.000	0.000
LOB_x	5.041e-05	0.000	0.328	0.744	-0.000	0.000
#P	0.0002	0.000	0.863	0.391	-0.000	0.001
PAge	-4.022e-05	0.001	-0.057	0.955	-0.001	0.001
RA/G_x	-0.0478	0.024	-1.987	0.050	-0.096	1.03e-05
ERA	-0.0389	0.027	-1.462	0.147	-0.092	0.014
G_y	-4.109e-05	1.3e-05	-3.170	0.002	-6.69e-05	-1.53e-05
GS_x	-4.109e-05	1.3e-05	-3.170	0.002	-6.69e-05	-1.53e-05
GF	1.081e-05	0.000	0.041	0.967	-0.001	0.001
CG_x	-5.191e-05	0.000	-0.197	0.844	-0.001	0.000
tSho	-0.0002	0.000	-0.828	0.410	-0.001	0.000
cSho	0.0003	0.001	0.279	0.781	-0.002	0.002
SV	2.642e-05	0.000	0.166	0.868	-0.000	0.000
IP	-0.0030	0.003	-0.982	0.329	-0.009	0.003
H_y	-0.0002	8.15e-05	-2.170	0.033	-0.000	-1.49e-05
R	0.0004	0.000	1.273	0.207	-0.000	0.001
ER	0.0002	0.000	0.981	0.329	-0.000	0.001
HR_y	0.0001	0.000	0.585	0.560	-0.000	0.001
BB_y	-0.0003	9.88e-05	-2.775	0.007	-0.000	-7.78e-05

IBB_y	-1.852e-05	7.62e-05	-0.243	0.809	-0.000	0.000
SO_y	0.0002	4.88e-05	3.532	0.001	7.53e-05	0.000
HBP_y	-0.0001	0.000	-0.891	0.376	-0.000	0.000
BK	-0.0002	0.000	-0.948	0.346	-0.001	0.000
WP	3.815e-05	5.62e-05	0.679	0.499	-7.35e-05	0.000
BF	0.0002	0.000	0.939	0.350	-0.000	0.001
ERA+	-0.0001	0.000	-0.586	0.559	-0.001	0.000
FIP	0.0074	0.026	0.286	0.776	-0.044	0.059
WHIP	0.3769	0.191	1.970	0.052	-0.003	0.757
H9	-0.0172	0.019	-0.899	0.371	-0.055	0.021
HR9	-0.0221	0.022	-0.992	0.324	-0.066	0.022
BB9	0.0118	0.019	0.624	0.534	-0.026	0.049
SO9	-0.0308	0.009	-3.625	0.000	-0.048	-0.014
SO/W	0.0148	0.012	1.187	0.238	-0.010	0.040
LOB_y	-0.0002	0.000	-1.025	0.308	-0.001	0.000
#Fld	0.0005	0.001	0.407	0.685	-0.002	0.003
RA/G_y	-0.0478	0.024	-1.987	0.050	-0.096	1.03e-05
DefEff	-0.1397	0.591	-0.236	0.814	-1.314	1.035
G	-4.109e-05	1.3e-05	-3.170	0.002	-6.69e-05	-1.53e-05
GS_y	-0.0004	0.000	-3.170	0.002	-0.001	-0.000
CG_y	1.102e-06	1.25e-05	0.088	0.930	-2.38e-05	2.6e-05
Inn	0.0003	0.001	0.528	0.599	-0.001	0.001
Ch	6.653e-05	0.000	0.219	0.827	-0.001	0.001
PO	-0.0002	0.001	-0.205	0.838	-0.002	0.002
A	-5.753e-05	0.000	-0.188	0.851	-0.001	0.001

```
E                    0.0003      0.000      0.990      0.325      -0.000
0.001
DP               -3.371e-05    6.8e-05     -0.496      0.621      -0.000
0.000
Fld%                 2.3292      0.865      2.694      0.008       0.611
4.048
Rtot                -0.0002      0.000     -1.613      0.110      -0.000
4.23e-05
Rtot/yr              0.0022      0.001      1.967      0.052   -2.34e-05
0.005
Rdrs             -1.125e-05   2.77e-05     -0.406      0.686   -6.63e-05
4.38e-05
Rdrs/yr              0.0006      0.000      1.351      0.180      -0.000
0.001
Rgood            -5.727e-05      0.000     -0.390      0.697      -0.000
0.000
teampayroll       2.562e-11   3.93e-11      0.651      0.517   -5.26e-11
1.04e-10
w                    0.0062      0.000     16.755      0.000       0.005
0.007
lastyrpayroll     2.765e-12   2.52e-11      0.110      0.913   -4.73e-11
5.28e-11
year                 0.0004      0.001      0.555      0.580      -0.001
0.002
perwin           -1.547e-09   1.71e-09     -0.904      0.368   -4.95e-09
1.85e-09
======================================================================
========
Omnibus:                       43.611   Durbin-Watson:
2.012
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
156.363
Skew:                           0.940   Prob(JB):
1.11e-34
Kurtosis:                       7.336   Cond. No.
1.06e+16
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The smallest eigenvalue is 5.83e-14. This might indicate that
there are
strong multicollinearity problems or that the design matrix is
singular.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
```

```
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)



MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)



print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Mean Square Error:

0.00029906733296893896

Root Mean Square Error:

0.017293563339258307
```

2nd Win percentage Model w/ pvalues

This model pulls all the significant pvalues from the previous model and puts those columns in our X, this preforms modereatly worse than our previous model with all of the columns, we get a adj R squared of .097 less than the model with all columns in it

```
model = df.copy()

X = model[['R_x', 'R/G', 'G_x', 'H_x', 'HR_x', 'BA', 'TB', 'RA/G_x',
'G_y', 'GS_x', 'BB_y', 'SO_y', 'WHIP', 'SO9', 'RA/G_y',
           'G', 'GS_y', 'Fld%', 'Rtot/yr']]
y = model['wpct']


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)

wpct_model2 = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = wpct_model2.fit()
results
```

```
results.rsquared

print(results.summary())
                              OLS Regression Results

======================================================================
========
Dep. Variable:                   wpct   R-squared:
0.908
Model:                            OLS   Adj. R-squared:
0.899
Method:                 Least Squares   F-statistic:
107.2
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
1.63e-71
Time:                        16:01:37   Log-Likelihood:
371.98
No. Observations:                 168   AIC:
-714.0
Df Residuals:                     153   BIC:
-667.1
Df Model:                          14

Covariance Type:            nonrobust

======================================================================
========
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
--------
const          0.1760      0.905      0.194      0.846      -1.612
1.964
R_x           -0.0108      0.074     -0.145      0.885      -0.158
0.136
R/G            0.1079      0.077      1.406      0.162      -0.044
0.259
G_x        -2.212e-05   1.17e-05     -1.883      0.062   -4.53e-05
1.09e-06
H_x            0.0003      0.000      1.825      0.070   -2.48e-05
0.001
HR_x           0.0005      0.000      1.753      0.082   -5.95e-05
0.001
BA            -1.0111      0.580     -1.743      0.083      -2.157
0.135
TB            -0.0001      0.000     -1.478      0.142      -0.000
5e-05
RA/G_x        -0.0477      0.005     -9.159      0.000      -0.058
-0.037
```

```
G_y         -2.212e-05   1.17e-05     -1.883      0.062   -4.53e-05
1.09e-06
GS_x        -2.212e-05   1.17e-05     -1.883      0.062   -4.53e-05
1.09e-06
BB_y        -2.628e-05   6.86e-05     -0.383      0.702      -0.000
0.000
SO_y            0.0001   8.43e-05      1.309      0.192   -5.62e-05
0.000
WHIP           -0.0068      0.080     -0.086      0.932      -0.164
0.151
SO9            -0.0186      0.013     -1.413      0.160      -0.045
0.007
RA/G_y         -0.0477      0.005     -9.159      0.000      -0.058
-0.037
G           -2.212e-05   1.17e-05     -1.883      0.062   -4.53e-05
1.09e-06
GS_y           -0.0002      0.000     -1.883      0.062      -0.000
9.82e-06
Fld%            0.7517      0.899      0.836      0.404      -1.025
2.528
Rtot/yr        -0.0002      0.001     -0.214      0.831      -0.002
0.002
======================================================================
========
Omnibus:                        2.821   Durbin-Watson:
2.305
Prob(Omnibus):                  0.244   Jarque-Bera (JB):
2.578
Skew:                           0.157   Prob(JB):
0.276
Kurtosis:                       3.519   Cond. No.
8.92e+21
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The smallest eigenvalue is 2.13e-35. This might indicate that
there are
strong multicollinearity problems or that the design matrix is
singular.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)
```

```
MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)



print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Mean Square Error:

0.0005712456696927833

Root Mean Square Error:

0.023900746216233143
```

Third Win Percentage Model w/ pvalues

```
model = df.copy()

X = model[['R_x', 'H_x', 'HR_x', 'BA', 'TB', 'RA/G_x', 'GS_x', 'BB_y',
'SO_y', 'WHIP', 'SO9',
            'Fld%', 'Rtot/yr']]
y = model['wpct']


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)

wpct_model2 = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = wpct_model2.fit()
results

results.rsquared

print(results.summary())
```

```
                            OLS Regression Results

==============================================================================
========
Dep. Variable:                   wpct   R-squared:
```

```
0.906
Model:                            OLS   Adj. R-squared:
0.898
Method:                 Least Squares   F-statistic:
114.6
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
3.95e-72
Time:                        16:01:37   Log-Likelihood:
370.91
No. Observations:                 168   AIC:
-713.8
Df Residuals:                     154   BIC:
-670.1
Df Model:                          13

Covariance Type:            nonrobust

======================================================================
========
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
--------
const          0.4326      0.889      0.486      0.627      -1.324
2.189
R_x            0.0926      0.012      7.896      0.000       0.069
0.116
H_x            0.0003      0.000      1.796      0.074   -2.96e-05
0.001
HR_x           0.0005      0.000      1.752      0.082   -5.99e-05
0.001
BA            -0.9412      0.580     -1.624      0.107      -2.086
0.204
TB            -0.0001      0.000     -1.397      0.164      -0.000
5.82e-05
RA/G_x        -0.0974      0.010     -9.422      0.000      -0.118
-0.077
GS_x          -0.0021      0.001     -2.091      0.038      -0.004
-0.000
BB_y       -3.064e-05   6.87e-05     -0.446      0.656      -0.000
0.000
SO_y           0.0001    8.4e-05      1.481      0.141   -4.15e-05
0.000
WHIP           0.0006      0.080      0.008      0.994      -0.157
0.158
SO9           -0.0208      0.013     -1.585      0.115      -0.047
0.005
Fld%           0.5145      0.886      0.581      0.562      -1.236
2.265
Rtot/yr       -0.0001      0.001     -0.112      0.911      -0.002
```

```
0.002
==============================================================================
========
Omnibus:                        3.823   Durbin-Watson:
2.324
Prob(Omnibus):                  0.148   Jarque-Bera (JB):
4.060
Skew:                           0.160   Prob(JB):
0.131
Kurtosis:                       3.691   Cond. No.
1.67e+06
==============================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The condition number is large, 1.67e+06. This might indicate that
there are
strong multicollinearity or other numerical problems.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)



MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)



print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Mean Square Error:

0.0005101306492817213

Root Mean Square Error:

0.02258607201975858
```

Third Win Percentage Model w/ correlation

```
model = df.copy()

X = model[['Rdiff', 'SRS', 'ERA+', 'ERA', 'RA/G_x']]
y = model['wpct']


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)

wpct_model3 = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = wpct_model3.fit()
results

results.rsquared

print(results.summary())
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:                   wpct   R-squared:
0.903
Model:                            OLS   Adj. R-squared:
0.900
Method:                 Least Squares   F-statistic:
302.5
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
3.33e-80
Time:                        16:01:37   Log-Likelihood:
368.19
No. Observations:                 168   AIC:
-724.4
Df Residuals:                     162   BIC:
-705.6
Df Model:                           5

Covariance Type:            nonrobust

======================================================================
========
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
--------
const          0.5350      0.063      8.533      0.000       0.411
0.659
```

```
Rdiff           0.0890      0.014      6.248      0.000       0.061
0.117
SRS             0.0001      0.014      0.009      0.993      -0.027
0.027
ERA+         7.284e-05      0.000      0.195      0.846      -0.001
0.001
ERA            -0.0319      0.021     -1.547      0.124      -0.073
0.009
RA/G_x          0.0205      0.020      1.013      0.313      -0.020
0.061
======================================================================
========
Omnibus:                        3.495   Durbin-Watson:
2.181
Prob(Omnibus):                  0.174   Jarque-Bera (JB):
3.154
Skew:                           0.233   Prob(JB):
0.207
Kurtosis:                       3.484   Cond. No.
3.06e+03
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The condition number is large, 3.06e+03. This might indicate that
there are
strong multicollinearity or other numerical problems.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)
```

```
MSE = np.square(np.subtract(y_test,y_pred)).mean()
RMSE = math.sqrt(MSE)
```

```
print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Root Mean Square Error:

0.025067914007983354
```

Payroll model

```
model = df.copy()

X = model.drop('teampayroll', axis=1)
y = model['teampayroll']

y = y.values.reshape(-1,1)


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)

salary_model = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = salary_model.fit()
results

results.rsquared

print(results.summary())

#cSho, R, ER, BB_y, RA/G_y
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:                      y   R-squared:
0.945
Model:                            OLS   Adj. R-squared:
0.895
Method:                 Least Squares   F-statistic:
18.76
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
8.47e-33
Time:                        16:01:37   Log-Likelihood:
-2966.4
No. Observations:                 168   AIC:
6095.
Df Residuals:                      87   BIC:
6348.
Df Model:                          80
```

```
Covariance Type:            nonrobust

==============================================================================
===========
                 coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
-----------
const       4.283e+09   5.09e+09      0.841      0.403   -5.84e+09
1.44e+10
R_x         2.578e+06   5.34e+07      0.048      0.962   -1.03e+08
1.09e+08
RA         -5.232e+07   6.32e+07     -0.828      0.410   -1.78e+08
7.33e+07
Rdiff       7.702e+07   6.11e+07      1.260      0.211   -4.45e+07
1.99e+08
SOS         3.614e+07   4.06e+07      0.890      0.376   -4.46e+07
1.17e+08
SRS        -1.199e+07   4.14e+07     -0.289      0.773   -9.44e+07
7.04e+07
#Bat        2.443e+06   3.08e+06      0.793      0.430   -3.68e+06
8.57e+06
BatAge      1.638e+06      2e+06      0.818      0.416   -2.34e+06
5.62e+06
R/G         -3.21e+07   1.07e+08     -0.299      0.766   -2.45e+08
1.81e+08
G_x          6.31e+04   3.66e+04      1.724      0.088   -9650.313
1.36e+05
PA         -6.052e+04   9.01e+05     -0.067      0.947   -1.85e+06
1.73e+06
AB           -1.5e+05   8.38e+05     -0.179      0.858   -1.82e+06
1.52e+06
R_y        -2.709e+05   5.73e+05     -0.472      0.638   -1.41e+06
8.69e+05
H_x         1.005e+05    2.9e+05      0.346      0.730   -4.76e+05
6.77e+05
2B         -8.785e+04   9.84e+04     -0.893      0.374   -2.83e+05
1.08e+05
3B         -1.341e+04   2.07e+05     -0.065      0.948   -4.25e+05
3.98e+05
HR_x        1.053e+05   1.37e+05      0.766      0.445   -1.68e+05
3.78e+05
RBI         1.299e+05   3.03e+05      0.429      0.669   -4.72e+05
7.32e+05
SB         -2.338e+04   9.19e+04     -0.255      0.800   -2.06e+05
1.59e+05
CS         -3.835e+05   3.71e+05     -1.034      0.304   -1.12e+06
3.53e+05
BB_x        2.879e+05    8.3e+05      0.347      0.729   -1.36e+06
1.94e+06
```

SO_x	4248.0548	2.56e+04	0.166	0.869	-4.66e+04	5.51e+04
BA	2.6e+09	1.81e+09	1.440	0.153	-9.89e+08	6.19e+09
OBP	1.966e+08	3.57e+09	0.055	0.956	-6.91e+09	7.3e+09
SLG	-1.526e+08	3.56e+09	-0.043	0.966	-7.22e+09	6.92e+09
OPS	-2.159e+09	3.46e+09	-0.624	0.534	-9.03e+09	4.71e+09
OPS+	8.105e+05	9.11e+05	0.890	0.376	-1e+06	2.62e+06
TB	3.018e+05	1.95e+05	1.545	0.126	-8.64e+04	6.9e+05
GDP	-3.271e+05	2.11e+05	-1.549	0.125	-7.47e+05	9.26e+04
HBP_x	3.452e+05	8.3e+05	0.416	0.679	-1.31e+06	2e+06
SH	7.331e+04	8.26e+05	0.089	0.930	-1.57e+06	1.72e+06
SF	-2.57e+05	9.23e+05	-0.279	0.781	-2.09e+06	1.58e+06
IBB_x	3.162e+05	2.35e+05	1.346	0.182	-1.51e+05	7.83e+05
LOB_x	-4.684e+04	4.18e+05	-0.112	0.911	-8.78e+05	7.84e+05
#P	3.64e+05	7.53e+05	0.483	0.630	-1.13e+06	1.86e+06
PAge	3.295e+06	1.88e+06	1.748	0.084	-4.51e+05	7.04e+06
RA/G_x	1.406e+08	6.52e+07	2.157	0.034	1.1e+07	2.7e+08
ERA	-2.007e+08	7e+07	-2.866	0.005	-3.4e+08	-6.15e+07
G_y	6.31e+04	3.66e+04	1.724	0.088	-9649.117	1.36e+05
GS_x	6.31e+04	3.66e+04	1.724	0.088	-9649.133	1.36e+05
GF	6.332e+05	7.05e+05	0.898	0.372	-7.68e+05	2.03e+06
CG_x	-5.701e+05	7.15e+05	-0.798	0.427	-1.99e+06	8.5e+05
tSho	-1.245e+05	6.59e+05	-0.189	0.851	-1.43e+06	1.19e+06
cSho	5.356e+06	2.58e+06	2.073	0.041	2.2e+05	1.05e+07
SV	6.267e+05	4.27e+05	1.467	0.146	-2.22e+05	1.48e+06
IP	1.918e+06	8.24e+06	0.233	0.816	-1.45e+07	1.83e+07

H_y	2.172e+04	2.27e+05	0.096	0.924	-4.3e+05	4.74e+05
R	-1.189e+06	7.51e+05	-1.584	0.117	-2.68e+06	3.03e+05
ER	1.363e+06	5.38e+05	2.532	0.013	2.93e+05	2.43e+06
HR_y	1.243e+05	5.98e+05	0.208	0.836	-1.07e+06	1.31e+06
BB_y	4.656e+05	2.76e+05	1.689	0.095	-8.22e+04	1.01e+06
IBB_y	-1.336e+05	2.07e+05	-0.646	0.520	-5.45e+05	2.77e+05
SO_y	-7.308e+04	1.42e+05	-0.516	0.607	-3.55e+05	2.08e+05
HBP_y	3.39e+04	3.09e+05	0.110	0.913	-5.8e+05	6.48e+05
BK	-6.268e+05	6.79e+05	-0.923	0.359	-1.98e+06	7.23e+05
WP	-2.395e+05	1.51e+05	-1.587	0.116	-5.39e+05	6.04e+04
BF	5.434e+04	6.11e+05	0.089	0.929	-1.16e+06	1.27e+06
ERA+	2.114e+05	6.81e+05	0.311	0.757	-1.14e+06	1.56e+06
FIP	-6.545e+07	7.06e+07	-0.927	0.356	-2.06e+08	7.49e+07
WHIP	-1.79e+08	5.31e+08	-0.337	0.737	-1.23e+09	8.77e+08
H9	6.453e+07	5.17e+07	1.248	0.215	-3.82e+07	1.67e+08
HR9	2.066e+07	6.08e+07	0.340	0.735	-1e+08	1.41e+08
BB9	-2.858e+05	5.16e+07	-0.006	0.996	-1.03e+08	1.02e+08
SO9	2.033e+07	2.47e+07	0.823	0.413	-2.88e+07	6.94e+07
SO/W	1.044e+07	3.41e+07	0.306	0.760	-5.74e+07	7.82e+07
LOB_y	-1.337e+05	5.85e+05	-0.228	0.820	-1.3e+06	1.03e+06
#Fld	-2.24e+06	3.13e+06	-0.717	0.476	-8.45e+06	3.97e+06
RA/G_y	1.406e+08	6.52e+07	2.157	0.034	1.1e+07	2.7e+08
DefEff	1.535e+09	1.6e+09	0.960	0.340	-1.64e+09	4.71e+09
G	6.31e+04	3.66e+04	1.724	0.088	-9649.124	1.36e+05
GS_y	5.679e+05	3.29e+05	1.724	0.088	-8.68e+04	1.22e+06

```
CG_y           1.553e+04    3.4e+04      0.457      0.649    -5.2e+04
8.31e+04
Inn           -8.039e+05   1.53e+06     -0.525      0.601   -3.85e+06
2.24e+06
Ch              2.92e+05   8.27e+05      0.353      0.725   -1.35e+06
1.94e+06
PO              1.28e+06   2.41e+06      0.530      0.597   -3.52e+06
6.08e+06
A             -2.586e+05   8.32e+05     -0.311      0.757   -1.91e+06
1.39e+06
E             -7.298e+05   8.42e+05     -0.867      0.389    -2.4e+06
9.44e+05
DP             6.181e+04   1.85e+05      0.334      0.739   -3.06e+05
4.29e+05
Fld%          -2.792e+09   2.43e+09     -1.150      0.253   -7.62e+09
2.03e+09
Rtot           2.395e+05   3.11e+05      0.770      0.443   -3.79e+05
8.58e+05
Rtot/yr       -4.144e+06   3.14e+06     -1.322      0.190   -1.04e+07
2.09e+06
Rdrs           4.095e+04   7.52e+04      0.544      0.588   -1.09e+05
1.9e+05
Rdrs/yr       -2.937e+05   1.19e+06     -0.246      0.806   -2.67e+06
2.08e+06
Rgood         -1.757e+05   3.99e+05     -0.441      0.661   -9.69e+05
6.17e+05
w             -1.081e+06   2.05e+06     -0.526      0.600   -5.16e+06
3e+06
wpct           1.893e+08   2.91e+08      0.651      0.517   -3.89e+08
7.67e+08
lastyrpayroll     0.3864      0.055      7.088      0.000       0.278
0.495
year          -1.188e+06   2.11e+06     -0.562      0.576   -5.39e+06
3.01e+06
perwin           33.6257      2.973     11.311      0.000      27.717
39.535
======================================================================
========
Omnibus:                        1.642   Durbin-Watson:
1.679
Prob(Omnibus):                  0.440   Jarque-Bera (JB):
1.327
Skew:                          -0.020   Prob(JB):
0.515
Kurtosis:                       3.434   Cond. No.
1.06e+16
======================================================================
========

Notes:
```

```
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 2.88e-14. This might indicate that there are

strong multicollinearity problems or that the design matrix is singular.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)



MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)



print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Mean Square Error:

607797450840487.8

Root Mean Square Error:

24653548.44318537
```

2nd Payroll Model w/pvalues

```
model = df.copy()

X = model[['cSho', 'R', 'ER', 'BB_y', 'RA/G_y']]
y = model['teampayroll']


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 42, test_size=.2)
```

```
salary_model = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = salary_model.fit()
results

results.rsquared

print(results.summary())
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:            teampayroll   R-squared:
0.105
Model:                            OLS   Adj. R-squared:
0.077
Method:                 Least Squares   F-statistic:
3.802
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
0.00278
Time:                        16:01:37   Log-Likelihood:
-3201.0
No. Observations:                 168   AIC:
6414.
Df Residuals:                     162   BIC:
6433.
Df Model:                           5

Covariance Type:            nonrobust

======================================================================
========
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
--------
const       2.425e+08    3.6e+07      6.734      0.000    1.71e+08
3.14e+08
cSho        4.983e+06   3.94e+06      1.263      0.208   -2.81e+06
1.28e+07
R          -3.632e+05   2.89e+05     -1.257      0.211   -9.34e+05
2.08e+05
ER          4.045e+05   3.02e+05      1.340      0.182   -1.92e+05
1e+06
BB_y       -1.742e+04   8.88e+04     -0.196      0.845   -1.93e+05
1.58e+05
RA/G_y     -2.441e+07   8.07e+06     -3.023      0.003   -4.03e+07   -
8.47e+06
======================================================================
```

```
========
Omnibus:                          5.580   Durbin-Watson:
1.733
Prob(Omnibus):                    0.061   Jarque-Bera (JB):
2.988
Skew:                            -0.006   Prob(JB):
0.224
Kurtosis:                         2.347   Cond. No.
1.10e+04
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The condition number is large, 1.1e+04. This might indicate that
there are
strong multicollinearity or other numerical problems.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)



MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)


print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Mean Square Error:

2359365589963762.0

Root Mean Square Error:

48573301.20512463
```

Third Salary model w/ correlations

```
model = df.copy()

X = model[['lastyrpayroll','PAge', 'BatAge', 'perwin', 'SRS']]
y = model['teampayroll']

y = y.values.reshape(-1,1)


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)

salary_model = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = salary_model.fit()
results

results.rsquared

print(results.summary())
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:                      y   R-squared:
0.777
Model:                            OLS   Adj. R-squared:
0.771
Method:                 Least Squares   F-statistic:
113.1
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
5.51e-51
Time:                        16:01:37   Log-Likelihood:
-3084.1
No. Observations:                 168   AIC:
6180.
Df Residuals:                     162   BIC:
6199.
Df Model:                           5

Covariance Type:            nonrobust

======================================================================
===========
                    coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
-----------
```

```
const          -3.33e+08   7.09e+07     -4.699      0.000   -4.73e+08
-1.93e+08
lastyrpayroll     0.6403      0.051     12.522      0.000       0.539
0.741
PAge           3.903e+06   2.03e+06      1.924      0.056   -1.03e+05
7.91e+06
BatAge         9.205e+06   1.94e+06      4.752      0.000    5.38e+06
1.3e+07
perwin            5.5014      1.658      3.319      0.001       2.228
8.775
SRS             1.22e+07   2.44e+06      5.004      0.000    7.39e+06
1.7e+07
======================================================================
========
Omnibus:                        0.117   Durbin-Watson:
2.037
Prob(Omnibus):                  0.943   Jarque-Bera (JB):
0.239
Skew:                          -0.049   Prob(JB):
0.888
Kurtosis:                       2.843   Cond. No.
5.49e+09
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The condition number is large, 5.49e+09. This might indicate that
there are
strong multicollinearity or other numerical problems.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)



MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)
```

```
print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Mean Square Error:

618069466864772.6

Root Mean Square Error:

24861002.93360613
```

Money per win Model

```
model = df.copy()

X = model.drop('perwin', axis=1)
y = model['perwin']

y = y.values.reshape(-1,1)


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)

salary_model = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = salary_model.fit()
results

results.rsquared

print(results.summary())
#G_x, TB, ERA, G_y, GS_x, R, ER, RA/G_y, Rtot/yr
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:                      y   R-squared:
0.957
Model:                            OLS   Adj. R-squared:
0.917
Method:                 Least Squares   F-statistic:
24.02
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
4.82e-37
Time:                        16:01:38   Log-Likelihood:
-2332.2
No. Observations:                 168   AIC:
```

```
4826.
Df Residuals:                      87   BIC:
5080.
Df Model:                          80

Covariance Type:            nonrobust

======================================================================
===========
                   coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
-----------
const        -1.766e+08   1.16e+08     -1.525      0.131   -4.07e+08
5.36e+07
R_x           -1.93e+05   1.22e+06     -0.158      0.875   -2.63e+06
2.24e+06
RA           -5.198e+05   1.45e+06     -0.357      0.722   -3.41e+06
2.37e+06
Rdiff        -2.039e+06    1.4e+06     -1.458      0.148   -4.82e+06
7.41e+05
SOS          -8.828e+05   9.31e+05     -0.948      0.346   -2.73e+06
9.69e+05
SRS           2.814e+05   9.51e+05      0.296      0.768   -1.61e+06
2.17e+06
#Bat         -6.724e+04   7.06e+04     -0.952      0.343   -2.08e+05
7.31e+04
BatAge        4.183e+04   4.59e+04      0.911      0.365   -4.94e+04
1.33e+05
R/G           1.541e+06   2.46e+06      0.627      0.532   -3.34e+06
6.43e+06
G_x          -1467.2832    839.447     -1.748      0.084   -3135.775
201.208
PA             606.9121   2.07e+04      0.029      0.977   -4.05e+04
4.17e+04
AB           -1328.0578   1.92e+04     -0.069      0.945   -3.96e+04
3.69e+04
R_y           1.162e+04   1.31e+04      0.886      0.378   -1.44e+04
3.77e+04
H_x           1597.7704   6655.783      0.240      0.811   -1.16e+04
1.48e+04
2B             338.8636   2267.833      0.149      0.882   -4168.700
4846.427
3B            -709.8068   4747.396     -0.150      0.881   -1.01e+04
8726.157
HR_x         -3098.9818   3146.240     -0.985      0.327   -9352.474
3154.511
RBI          -5088.8070   6938.306     -0.733      0.465   -1.89e+04
8701.828
SB            -634.9503   2107.237     -0.301      0.764   -4823.313
```

```
3553.412
CS           5403.5906   8539.474      0.633      0.529   -1.16e+04
2.24e+04
BB_x        -1.108e+04    1.9e+04     -0.583      0.562   -4.89e+04
2.67e+04
SO_x         -251.9166    586.930     -0.429      0.669   -1418.503
914.669
BA          -7.278e+07   4.12e+07     -1.767      0.081   -1.55e+08
9.06e+06
OBP         -3.139e+06    8.2e+07     -0.038      0.970   -1.66e+08
1.6e+08
SLG          1.643e+07   8.16e+07      0.201      0.841   -1.46e+08
1.79e+08
OPS          4.658e+07   7.94e+07      0.587      0.559   -1.11e+08
2.04e+08
OPS+          -1.86e+04   2.09e+04     -0.890      0.376   -6.01e+04
2.29e+04
TB          -8779.9116   4443.112     -1.976      0.051   -1.76e+04
51.255
GDP          4489.5976   4887.968      0.918      0.361   -5225.768
1.42e+04
HBP_x       -1.254e+04    1.9e+04     -0.659      0.511   -5.04e+04
2.53e+04
SH          -1.045e+04   1.89e+04     -0.552      0.582   -4.81e+04
2.72e+04
SF             25.4767   2.12e+04      0.001      0.999   -4.21e+04
4.21e+04
IBB_x        -608.4209   5446.378     -0.112      0.911   -1.14e+04
1.02e+04
LOB_x        5813.2499   9573.677      0.607      0.545   -1.32e+04
2.48e+04
#P           2332.2055   1.73e+04      0.135      0.893   -3.21e+04
3.67e+04
PAge        -4.553e+04   4.37e+04     -1.041      0.301   -1.32e+05
4.14e+04
RA/G_x      -2.804e+06    1.5e+06     -1.863      0.066    -5.8e+06
1.87e+05
ERA          6.067e+06   1.55e+06      3.913      0.000    2.99e+06
9.15e+06
G_y         -1467.2853    839.444     -1.748      0.084   -3135.771
201.201
GS_x        -1467.2856    839.444     -1.748      0.084   -3135.772
201.200
GF          -1.514e+04   1.62e+04     -0.936      0.352   -4.73e+04
1.7e+04
CG_x         1.367e+04   1.64e+04      0.834      0.406   -1.89e+04
4.63e+04
tSho         3547.0777   1.51e+04      0.235      0.815   -2.65e+04
3.36e+04
cSho        -8.367e+04   6.01e+04     -1.393      0.167   -2.03e+05
```

						3.57e+04
SV	-1.084e+04	9852.385	-1.100	0.274	-3.04e+04	8747.416
IP	5.794e+04	1.89e+05	0.307	0.760	-3.18e+05	4.34e+05
H_y	3589.1826	5202.679	0.690	0.492	-6751.705	1.39e+04
R	2.466e+04	1.73e+04	1.428	0.157	-9663.238	5.9e+04
ER	-4.33e+04	1.19e+04	-3.630	0.000	-6.7e+04	-1.96e+04
HR_y	1528.3518	1.37e+04	0.111	0.912	-2.58e+04	2.88e+04
BB_y	-6326.7442	6390.532	-0.990	0.325	-1.9e+04	6375.131
IBB_y	3371.4944	4741.756	0.711	0.479	-6053.259	1.28e+04
SO_y	767.8652	3252.612	0.236	0.814	-5697.053	7232.783
HBP_y	767.8729	7085.133	0.108	0.914	-1.33e+04	1.49e+04
BK	8235.0552	1.56e+04	0.527	0.600	-2.28e+04	3.93e+04
WP	4726.8730	3474.873	1.360	0.177	-2179.813	1.16e+04
BF	5346.3114	1.4e+04	0.382	0.704	-2.25e+04	3.32e+04
ERA+	-1.384e+04	1.56e+04	-0.889	0.376	-4.48e+04	1.71e+04
FIP	6.2e+05	1.63e+06	0.381	0.704	-2.61e+06	3.85e+06
WHIP	1.315e+07	1.21e+07	1.086	0.281	-1.09e+07	3.72e+07
H9	-1.717e+06	1.18e+06	-1.452	0.150	-4.07e+06	6.33e+05
HR9	-9.619e+05	1.39e+06	-0.691	0.491	-3.73e+06	1.8e+06
BB9	-9.054e+05	1.18e+06	-0.768	0.445	-3.25e+06	1.44e+06
SO9	2.462e+04	5.69e+05	0.043	0.966	-1.11e+06	1.16e+06
SO/W	-1.066e+05	7.83e+05	-0.136	0.892	-1.66e+06	1.45e+06
LOB_y	-5780.4382	1.34e+04	-0.431	0.668	-3.24e+04	2.09e+04
#Fld	5.437e+04	7.17e+04	0.758	0.450	-8.82e+04	1.97e+05
RA/G_y	-2.804e+06	1.5e+06	-1.863	0.066	-5.8e+06	1.87e+05
DefEff	5.661e+06	3.69e+07	0.154	0.878	-6.76e+07	

```
7.89e+07
G              -1467.2856    839.444     -1.748      0.084   -3135.772
201.200
GS_y           -1.321e+04   7554.982     -1.748      0.084   -2.82e+04
1810.751
CG_y             121.8968    780.823      0.156      0.876   -1430.074
1673.868
Inn             9971.1811   3.52e+04      0.283      0.778   -5.99e+04
7.99e+04
Ch             -7468.4625    1.9e+04     -0.394      0.695   -4.52e+04
3.02e+04
PO             -3.791e+04   5.53e+04     -0.685      0.495   -1.48e+05
7.21e+04
A               7132.2776   1.91e+04      0.374      0.709   -3.08e+04
4.51e+04
E               2.331e+04   1.92e+04      1.211      0.229   -1.49e+04
6.16e+04
DP              -720.9978   4244.267     -0.170      0.866   -9156.939
7714.943
Fld%            6.219e+07   5.57e+07      1.116      0.268   -4.86e+07
1.73e+08
Rtot           -1.082e+04   7068.763     -1.530      0.130   -2.49e+04
3231.915
Rtot/yr         1.452e+05    7.1e+04      2.046      0.044    4134.177
2.86e+05
Rdrs           -1023.4975   1725.744     -0.593      0.555   -4453.600
2406.605
Rdrs/yr         1.045e+04   2.74e+04      0.381      0.704    -4.4e+04
6.49e+04
Rgood           7273.6297   9128.852      0.797      0.428   -1.09e+04
2.54e+04
teampayroll        0.0177      0.002     11.311      0.000       0.015
0.021
w               1.598e+04   4.72e+04      0.339      0.736   -7.78e+04
1.1e+05
wpct           -6.018e+06   6.66e+06     -0.904      0.368   -1.92e+07
7.21e+06
lastyrpayroll     -0.0028      0.002     -1.796      0.076      -0.006
0.000
year            4.648e+04   4.83e+04      0.962      0.339   -4.95e+04
1.42e+05
======================================================================
========
Omnibus:                       32.928   Durbin-Watson:
1.775
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
325.493
Skew:                          -0.010   Prob(JB):
2.09e-71
Kurtosis:                       9.819   Cond. No.
```

```
1.06e+16
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 5.83e-14. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)



MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)



print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Mean Square Error:

226112208558.079

Root Mean Square Error:

475512.57455305953
```

2nd per win model w/ pvalues

```
model = df.copy()

X = model[['G_x', 'TB', 'ERA', 'G_y', 'GS_x', 'R', 'ER', 'RA/G_y', 'Rtot/yr']]
y = model['perwin']
```

```
y = y.values.reshape(-1,1)


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)

salary_model = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = salary_model.fit()
results

results.rsquared

print(results.summary())
                            OLS Regression Results

======================================================================
========
Dep. Variable:                      y   R-squared:
0.627
Model:                            OLS   Adj. R-squared:
0.611
Method:                 Least Squares   F-statistic:
38.48
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
2.93e-31
Time:                        16:01:38   Log-Likelihood:
-2513.0
No. Observations:                 168   AIC:
5042.
Df Residuals:                     160   BIC:
5067.
Df Model:                           7

Covariance Type:            nonrobust

======================================================================
========
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
--------
const       2.954e+06       2e+06      1.474      0.142      -1e+06
6.91e+06
G_x        -4588.6531   4801.036     -0.956      0.341   -1.41e+04
4892.921
TB           581.6716    397.933      1.462      0.146    -204.207
1367.550
ERA         1.628e+06   1.05e+06      1.546      0.124   -4.51e+05
```

```
3.71e+06
G_y         -4588.6531   4801.036     -0.956      0.341   -1.41e+04
4892.921
GS_x        -4588.6531   4801.036     -0.956      0.341   -1.41e+04
4892.921
R             232.1517   8959.798      0.026      0.979   -1.75e+04
1.79e+04
ER          -4906.9544   9108.608     -0.539      0.591   -2.29e+04
1.31e+04
RA/G_y      -9.096e+05   1.09e+06     -0.833      0.406   -3.07e+06
1.25e+06
Rtot/yr     -3.982e+04   2.14e+04     -1.863      0.064    -8.2e+04
2401.258
======================================================================
========
Omnibus:                       12.312   Durbin-Watson:
1.884
Prob(Omnibus):                  0.002   Jarque-Bera (JB):
29.044
Skew:                           0.189   Prob(JB):
4.93e-07
Kurtosis:                       5.002   Cond. No.
1.30e+20
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The smallest eigenvalue is 5.52e-32. This might indicate that
there are
strong multicollinearity problems or that the design matrix is
singular.
```

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)



MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)
```

```
print('\nRoot Mean Square Error:\n')
print(RMSE)
```

```
Mean Square Error:

579787888871.3977

Root Mean Square Error:

761438.040073779
```

3rd per win model with correlations

```
model = df.copy()

X = model[['GF', 'Ch', 'PO', 'IP', 'GS_y']]
y = model['perwin']

y = y.values.reshape(-1,1)


X_train, X_test, y_train, y_test = train_test_split(X, y, random_state
= 42, test_size=.2)

salary_model = sm.OLS(endog=y_train, exog=sm.add_constant(X_train))

results = salary_model.fit()
results

results.rsquared

print(results.summary())
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:                      y   R-squared:
0.608
Model:                            OLS   Adj. R-squared:
0.595
Method:                 Least Squares   F-statistic:
50.15
Date:                Wed, 16 Nov 2022   Prob (F-statistic):
3.39e-31
Time:                        16:01:38   Log-Likelihood:
-2517.4
No. Observations:                 168   AIC:
```

```
5047.
Df Residuals:                     162   BIC:
5066.
Df Model:                           5

Covariance Type:            nonrobust

======================================================================
========
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
--------
const       5.855e+06   2.87e+05     20.366      0.000    5.29e+06
6.42e+06
GF         -6.413e+04   3.74e+04     -1.713      0.089   -1.38e+05
9815.568
Ch          -640.6168    603.482     -1.062      0.290   -1832.323
551.089
PO         -1.335e+05   7.25e+04     -1.841      0.068   -2.77e+05
9733.262
IP          4.001e+05   2.17e+05      1.840      0.068   -2.93e+04
8.29e+05
GS_y        7350.6776   6072.367      1.211      0.228   -4640.521
1.93e+04
======================================================================
========
Omnibus:                        8.555   Durbin-Watson:
1.892
Prob(Omnibus):                  0.014   Jarque-Bera (JB):
16.652
Skew:                           0.085   Prob(JB):
0.000242
Kurtosis:                       4.533   Cond. No.
3.34e+04
======================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The condition number is large, 3.34e+04. This might indicate that
there are
strong multicollinearity or other numerical problems.
```

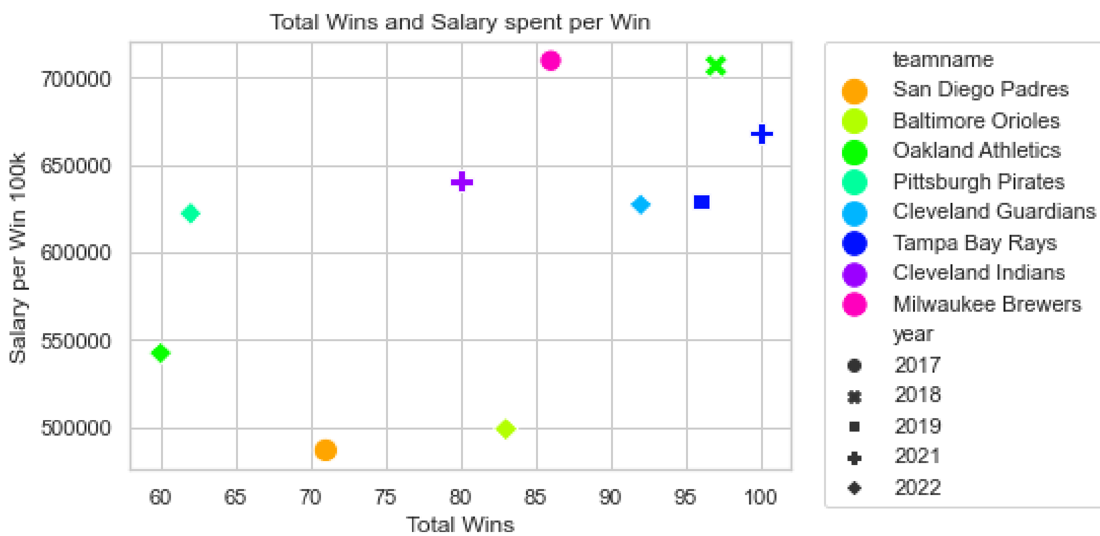
```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.transform(X_train)
regressor = LinearRegression()
reg = regressor.fit(X_train, y_train)
```

```
reg.score(X_test, y_test)
y_pred = reg.predict(X_test)



MSE = np.square(np.subtract(y_test,y_pred)).mean()

RMSE = math.sqrt(MSE)
print('Mean Square Error:\n')
print(MSE)

print('\nRoot Mean Square Error:\n')
print(RMSE)
```

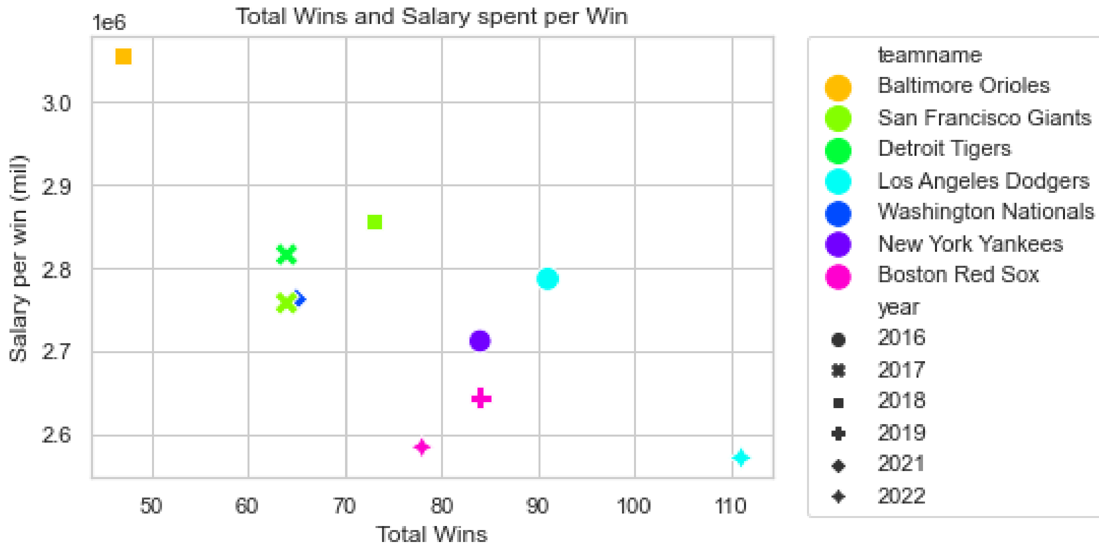
```
Mean Square Error:

607773445842.5756

Root Mean Square Error:

779598.259260868
```

Visualizations

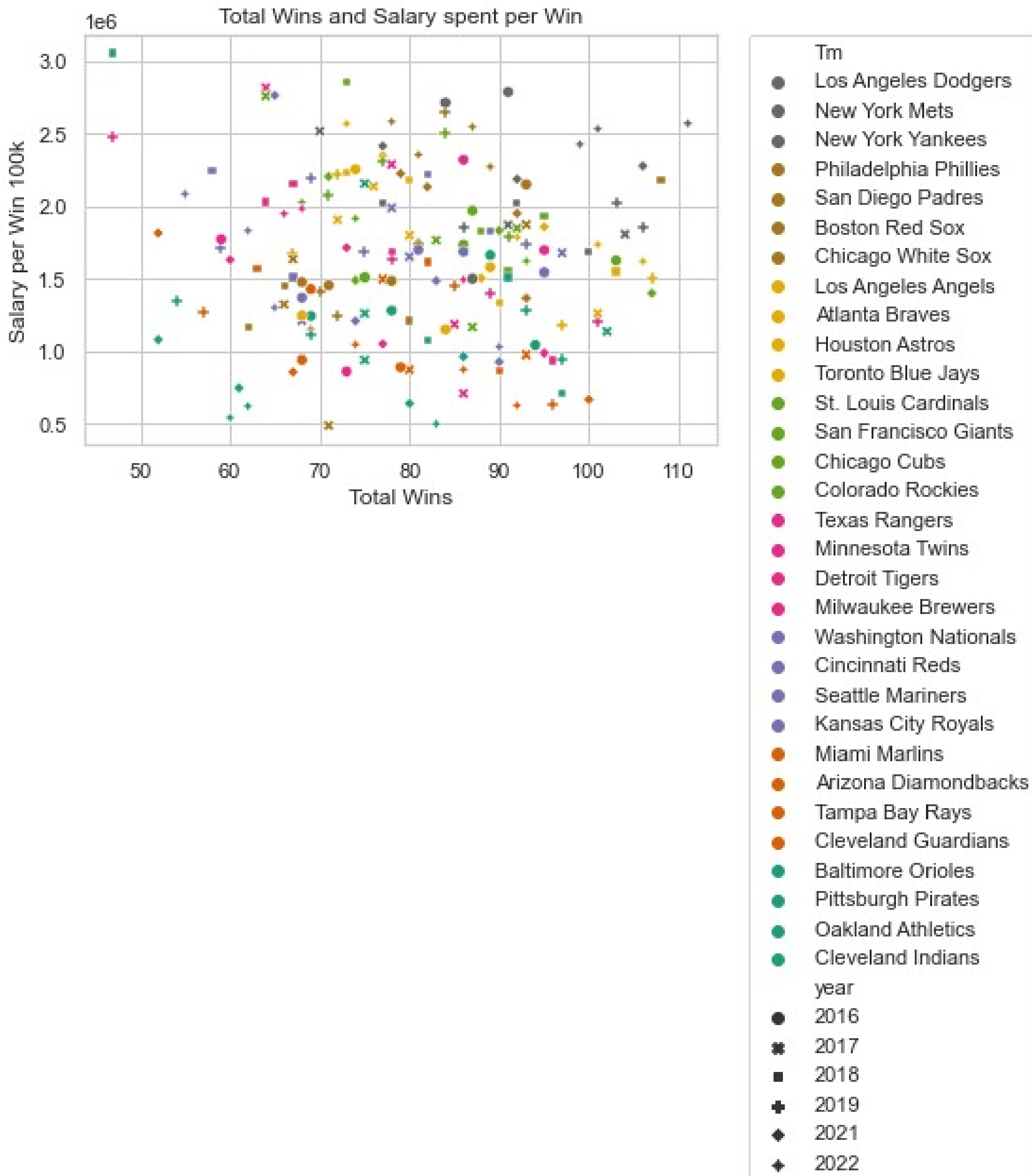
```
sns.set_theme(style="whitegrid", palette='copper_r')
sns.scatterplot(data=df_salary.sort_values(by=['perwin'],
ascending=True)[:10], x="w", y="perwin",hue='teamname', style ='year',
                size='teamname', sizes=(130,150), palette = 'hsv');
plt.xlabel('Total Wins')
plt.ylabel('Salary per Win 100k')
plt.title('Total Wins and Salary spent per Win');
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.);
```



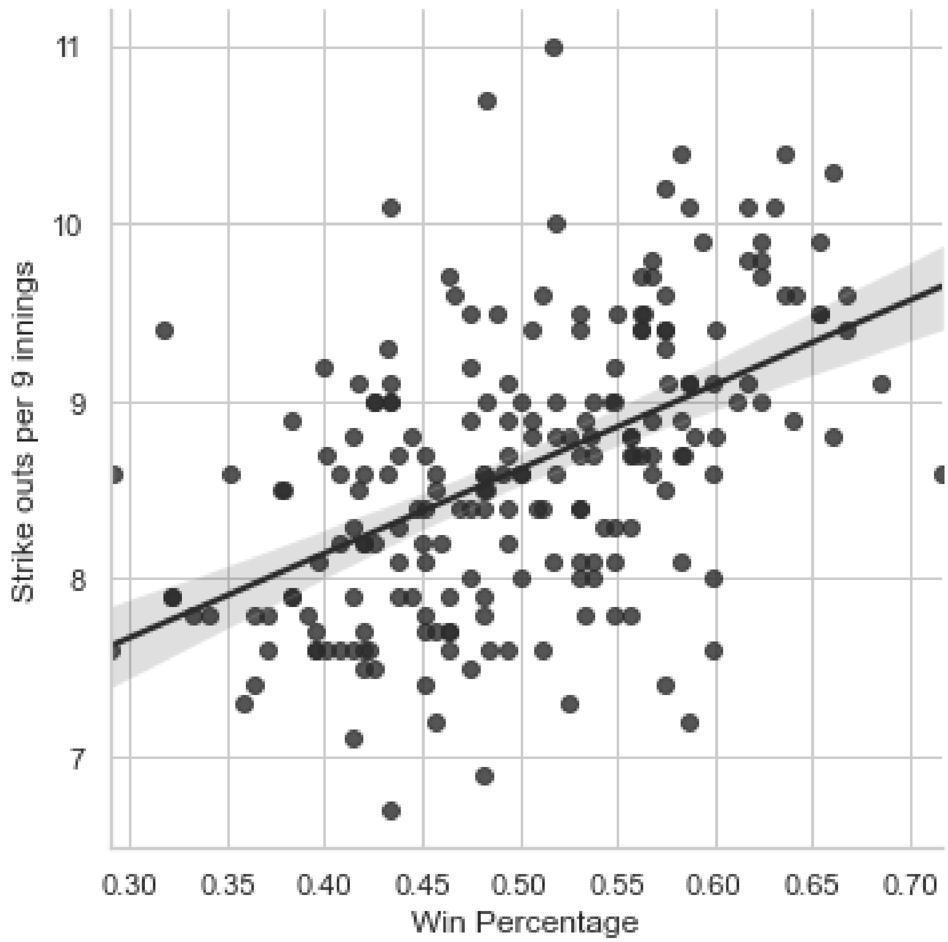
```
sns.set_theme(style="whitegrid", palette='copper_r')
sns.scatterplot(data=df_salary.sort_values(by=['perwin'],
ascending=False)[:10], x="w", y="perwin",hue='teamname', style
='year',
                    size='teamname', sizes=(130,150), palette = 'hsv');
plt.xlabel('Total Wins')
plt.ylabel('Salary per Win (mil)')
plt.title('Total Wins and Salary spent per Win');
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.);
```



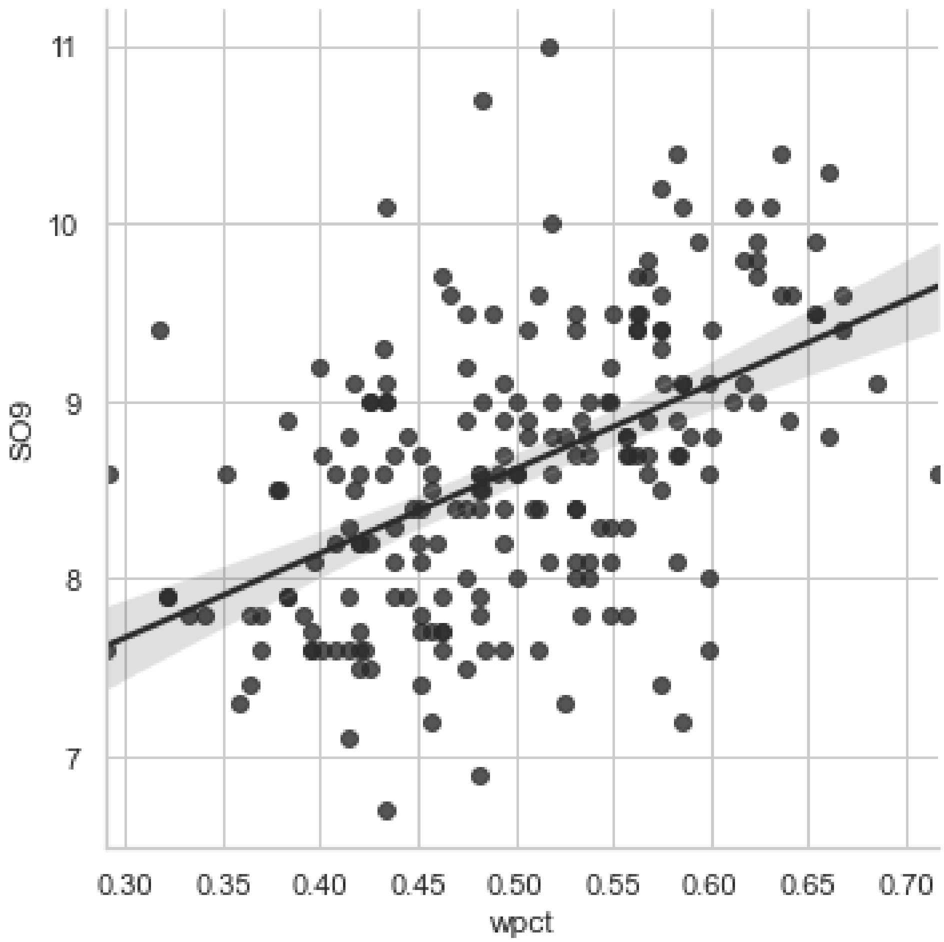

```
sns.scatterplot(data=df_salary, x="w", y="perwin", hue ='Tm',
style='year', palette= 'Dark2_r');
plt.xlabel('Total Wins')
plt.ylabel('Salary per Win 100k')
plt.title('Total Wins and Salary spent per Win');
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.);
```



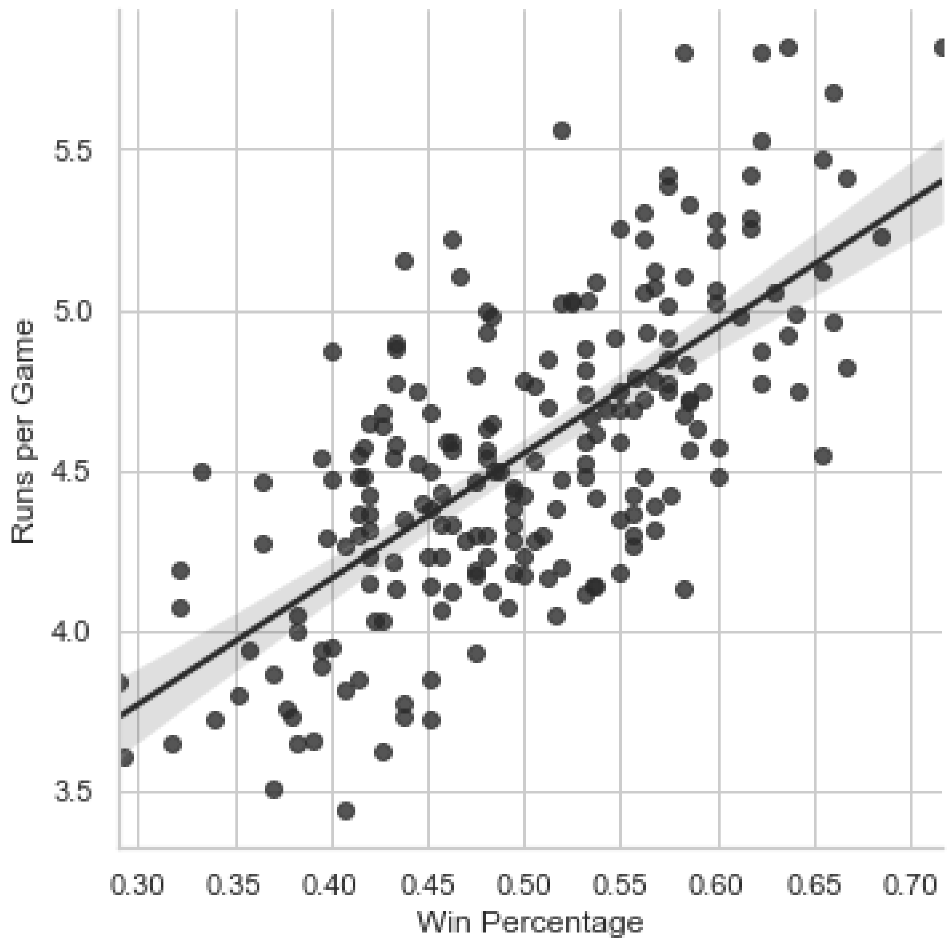

```
sns.set_theme(style="whitegrid", palette='Greys_r')
sns.lmplot(data=df, x= 'wpct', y= 'SO9')
plt.xlabel('Win Percentage')
plt.ylabel('Strike outs per 9 innings');
```



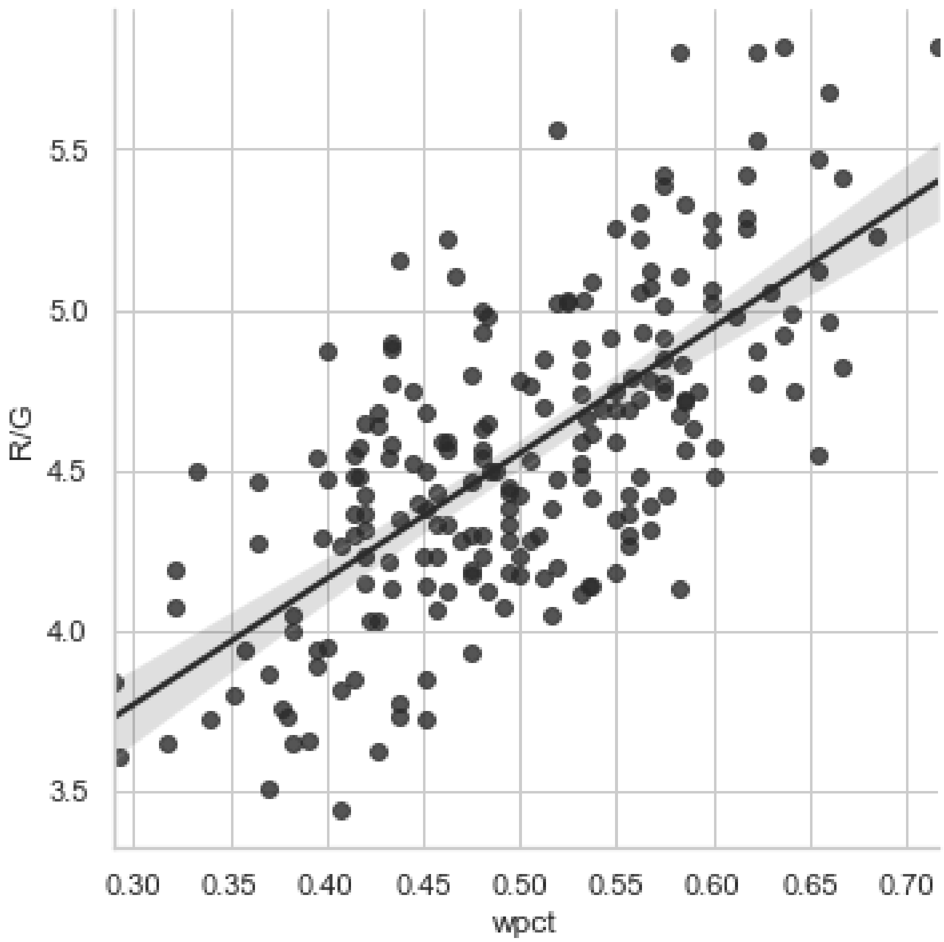

```
sns.set_theme(style="whitegrid", palette='Greys_r')
sns.lmplot(data=df, x= 'wpct', y= 'SO9');
```




```
sns.set_theme(style="whitegrid", palette='Greys_r')
sns.lmplot(data=df, x= 'wpct', y= 'R/G')
plt.xlabel('Win Percentage')
plt.ylabel('Runs per Game');
```




```
sns.set_theme(style="whitegrid", palette='Greys_r')
sns.lmplot(data=df, x= 'wpct', y= 'R/G');
```



R/G
wpct
5.5
5.0
4.5
4.0
3.5
0.30
0.35
0.40
0.45
0.50
0.55
0.60
0.65
0.70