Brief description of the data set and a summary of its attributes

This dataset is about the performance of basketball teams. The **basketball_train.csv** data set includes performance data about five seasons of 355 basketball teams. It includes following fields:

Field	Description
TEAM	The Division I college basketball school
CONF	The Athletic Conference in which the school participates in (A10 = Atlantic 10, ACC = Atlantic Coast Conference, AE = America East, Amer = American, ASun = ASUN, B10 = Big Ten, B12 = Big 12, BE = Big East, BSky = Big Sky, BSth = Big South, BW = Big West, CAA = Colonial Athletic Association, CUSA = Conference USA, Horz = Horizon League, Ivy = Ivy League, MAAC = Metro Atlantic Athletic Conference, MAC = Mid-American Conference, MEAC = Mid-Eastern Athletic Conference, MVC = Missouri Valley Conference, MWC = Mountain West, NEC = Northeast Conference, OVC = Ohio Valley Conference, P12 = Pac-12, Pat = Patriot League, SB = Sun Belt, SC = Southern Conference, SEC = South Eastern Conference, Slnd = Southland Conference, Sum = Summit League, SWAC = Southwestern Athletic Conference, WAC = Western Athletic Conference, WCC = West Coast Conference)
G	Number of games played
W	Number of games won
ADJOE	Adjusted Offensive Efficiency (An estimate of the offensive efficiency (points scored per 100 possessions) a team would have against the average Division I defense)
ADJDE	Adjusted Defensive Efficiency (An estimate of the defensive efficiency (points allowed per 100 possessions) a team would have against the average Division I offense)
BARTHAG	Power Rating (Chance of beating an average Division I team)
EFG_O	Effective Field Goal Percentage Shot
EFG_D	Effective Field Goal Percentage Allowed
TOR	Turnover Percentage Allowed (Turnover Rate)
TORD	Turnover Percentage Committed (Steal Rate)
ORB	Offensive Rebound Percentage
DRB	Defensive Rebound Percentage
FTR	Free Throw Rate (How often the given team shoots Free Throws)
FTRD	Free Throw Rate Allowed
2P_O	Two-Point Shooting Percentage
2P_D	Two-Point Shooting Percentage Allowed
3P_O	Three-Point Shooting Percentage
3P_D	Three-Point Shooting Percentage Allowed
ADJ_T	Adjusted Tempo (An estimate of the tempo (possessions per 40 minutes) a team would have against the team that wants to play at an average Division I tempo)
WAB	Wins Above Bubble (The bubble refers to the cut off between making the NCAA March Madness Tournament and not making it)
POSTSEASON	Round where the given team was eliminated or where their season ended (R68 = First Four, R64 = Round of 64, R32 = Round of 32, S16 = Sweet Sixteen, E8 = Elite Eight, F4 = Final Four, 2ND = Runner-up, Champion = Winner of the NCAA March Madness Tournament for that given year)
SEED	Seed in the NCAA March Madness Tournament
YEAR	Season

Source: NCAA Division I Men's Basketball Tournament (https://en.wikipedia.org/wiki/NCAA_Division_I_Men%27s_Basketball_Tournament (https://en.wikipedia.org/wiki/NCAA_Division_I_Men%27s_Basketball_Tournament))

Initial plan for data exploration

The basketball dataset is downloaded from relevant website. We will plan to analyse what factors that made a team successfully made to **Final Four (SemiFinals)**. We need to examine the features and find any data patterns within the dataset. There will be some visualizations done and a hypothesis testing is conducted.

Actions taken for data cleaning and feature engineering

As for data cleaning, we will check for missing values and decide what imputation method. We also check for data duplicates and outliers. Finally perform binary encoding before model training.

Key Findings and Insights, which synthesizes the results of Exploratory Data Analysis in an insightful and actionable manner

Import Libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from scipy import stats
from scipy.stats import pearsonr
import math


%matplotlib inline
sns.set_style('dark')
sns.set(font_scale=1.2)

import warnings
warnings.filterwarnings('ignore')

np.random.seed(123)

pd.options.display.max_columns= None
#pd.options.display.max_rows = None
```

In [2]:
```
df = pd.read_csv("basketball_train.csv")
```

In [3]:
```
df
```

Out[3]:

	TEAM	CONF	G	W	ADJOE	ADJDE	BARTHAG	EFG_O	EFG_D	TOR	TORD	ORB	DRB	FTR	F
0	North Carolina	ACC	40	33	123.3	94.9	0.9531	52.6	48.1	15.4	18.2	40.7	30.0	32.3	
1	Villanova	BE	40	35	123.1	90.9	0.9703	56.1	46.7	16.3	20.6	28.2	29.4	34.1	
2	Notre Dame	ACC	36	24	118.3	103.3	0.8269	54.0	49.5	15.3	14.8	32.7	32.1	32.9	
3	Virginia	ACC	37	29	119.9	91.0	0.9600	54.8	48.4	15.1	18.8	29.9	25.2	32.1	
4	Kansas	B12	37	32	120.9	90.4	0.9662	55.7	45.1	17.8	18.5	32.2	27.9	38.6	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1752	UCLA	P12	36	22	111.8	96.6	0.8425	49.6	48.5	17.6	17.9	33.8	28.6	35.7	
1753	Utah	P12	34	25	114.9	88.7	0.9513	55.2	43.0	18.2	18.3	31.3	28.4	43.4	
1754	West Virginia	B12	35	25	110.3	93.3	0.8733	46.1	52.7	18.7	28.0	40.1	31.1	40.4	
1755	Wichita St.	MVC	34	29	114.3	91.5	0.9277	50.3	45.8	15.0	21.3	34.5	27.4	36.2	
1756	Xavier	BE	37	23	115.7	95.1	0.9049	53.3	50.0	18.1	18.8	31.3	27.3	38.5	

1757 rows × 24 columns

Dataset has 5 categorical features and 19 numeric features

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1757 entries, 0 to 1756
Data columns (total 24 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   TEAM        1757 non-null   object 
 1   CONF        1757 non-null   object 
 2   G           1757 non-null   int64  
 3   W           1757 non-null   int64  
 4   ADJOE       1757 non-null   float64
 5   ADJDE       1757 non-null   float64
 6   BARTHAG     1757 non-null   float64
 7   EFG_O       1757 non-null   float64
 8   EFG_D       1757 non-null   float64
 9   TOR         1757 non-null   float64
 10  TORD        1757 non-null   float64
 11  ORB         1757 non-null   float64
 12  DRB         1757 non-null   float64
 13  FTR         1757 non-null   float64
 14  FTRD        1757 non-null   float64
 15  2P_O        1757 non-null   float64
 16  2P_D        1757 non-null   float64
 17  3P_O        1757 non-null   float64
 18  3P_D        1757 non-null   float64
 19  ADJ_T       1757 non-null   float64
 20  WAB         1757 non-null   float64
 21  POSTSEASON  1757 non-null   object 
 22  SEED        1757 non-null   int64  
 23  YEAR        1757 non-null   int64  
dtypes: float64(17), int64(4), object(3)
memory usage: 329.6+ KB
```

Summary of statistics below:

In [5]:
```
df.describe(include='all').T
```

Out[5]:

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
TEAM	1757	355	Creighton	5	NaN	NaN	NaN	NaN	NaN	NaN	NaN
CONF	1757	33	ACC	75	NaN	NaN	NaN	NaN	NaN	NaN	NaN
G	1757	NaN	NaN	NaN	31.5231	2.60282	24	30	31	33	40
W	1757	NaN	NaN	NaN	16.5134	6.54557	0	12	16	21	38
ADJOE	1757	NaN	NaN	NaN	103.542	7.30498	76.7	98.6	103.1	108.1	129.1
ADJDE	1757	NaN	NaN	NaN	103.542	6.47268	84	98.9	103.8	108	124
BARTHAG	1757	NaN	NaN	NaN	0.493398	0.255291	0.0077	0.2837	0.474	0.7106	0.9842
EFG_O	1757	NaN	NaN	NaN	50.1205	3.13043	39.4	48.1	50	52.1	59.8
EFG_D	1757	NaN	NaN	NaN	50.3128	2.8596	39.6	48.4	50.3	52.3	59.5
TOR	1757	NaN	NaN	NaN	18.5918	1.99164	12.4	17.2	18.5	19.8	26.1
TORD	1757	NaN	NaN	NaN	18.5213	2.10897	10.2	17.1	18.5	19.9	28
ORB	1757	NaN	NaN	NaN	29.2771	4.10178	15	26.6	29.4	31.9	42.1
DRB	1757	NaN	NaN	NaN	29.4674	3.06179	18.4	27.3	29.4	31.5	40.4
FTR	1757	NaN	NaN	NaN	35.0979	4.8846	21.6	31.7	34.9	38.3	51
FTRD	1757	NaN	NaN	NaN	35.3733	5.90094	21.8	31.2	34.9	39.2	58.5
2P_O	1757	NaN	NaN	NaN	49.136	3.42214	37.7	46.9	49	51.4	62.6
2P_D	1757	NaN	NaN	NaN	49.2981	3.28826	37.7	47.1	49.3	51.5	61.2
3P_O	1757	NaN	NaN	NaN	34.5635	2.74232	25.2	32.6	34.6	36.4	44.1
3P_D	1757	NaN	NaN	NaN	34.7448	2.36973	27.1	33.1	34.7	36.3	43.1
ADJ_T	1757	NaN	NaN	NaN	68.4223	3.25892	57.2	66.4	68.5	70.4	83.4
WAB	1757	NaN	NaN	NaN	-7.83711	6.98869	-25.2	-13	-8.4	-3.1	13.1
POSTSEASON	1757	9	N	1417	NaN	NaN	NaN	NaN	NaN	NaN	NaN
SEED	1757	NaN	NaN	NaN	1.7012	4.03559	0	0	0	0	16
YEAR	1757	NaN	NaN	NaN	2017	1.41542	2015	2016	2017	2018	2019

Shape of dataset:

In [6]:
```
df.shape
```

Out[6]:
```
(1757, 24)
```

In [7]:
```
df.columns
```

Out[7]:
```
Index(['TEAM', 'CONF', 'G', 'W', 'ADJOE', 'ADJDE', 'BARTHAG', 'EFG_O', 'EFG_D',
       'TOR', 'TORD', 'ORB', 'DRB', 'FTR', 'FTRD', '2P_O', '2P_D', '3P_O',
       '3P_D', 'ADJ_T', 'WAB', 'POSTSEASON', 'SEED', 'YEAR'],
      dtype='object')
```

Data Exploration

Total 355 teams in dataset

In [8]:
```
df['TEAM'].nunique()
```
Out[8]:
```
355
```

Teams appears in each season:

In [9]:
```
df['TEAM'].value_counts()
```
Out[9]:
```
Creighton               5
UCF                     5
Oregon St.              5
Kansas St.              5
New Orleans             5
                       ..
Fort Wayne              3
Arkansas Little Rock    2
IPFW                    2
North Alabama           1
Cal Baptist             1
Name: TEAM, Length: 355, dtype: int64
```

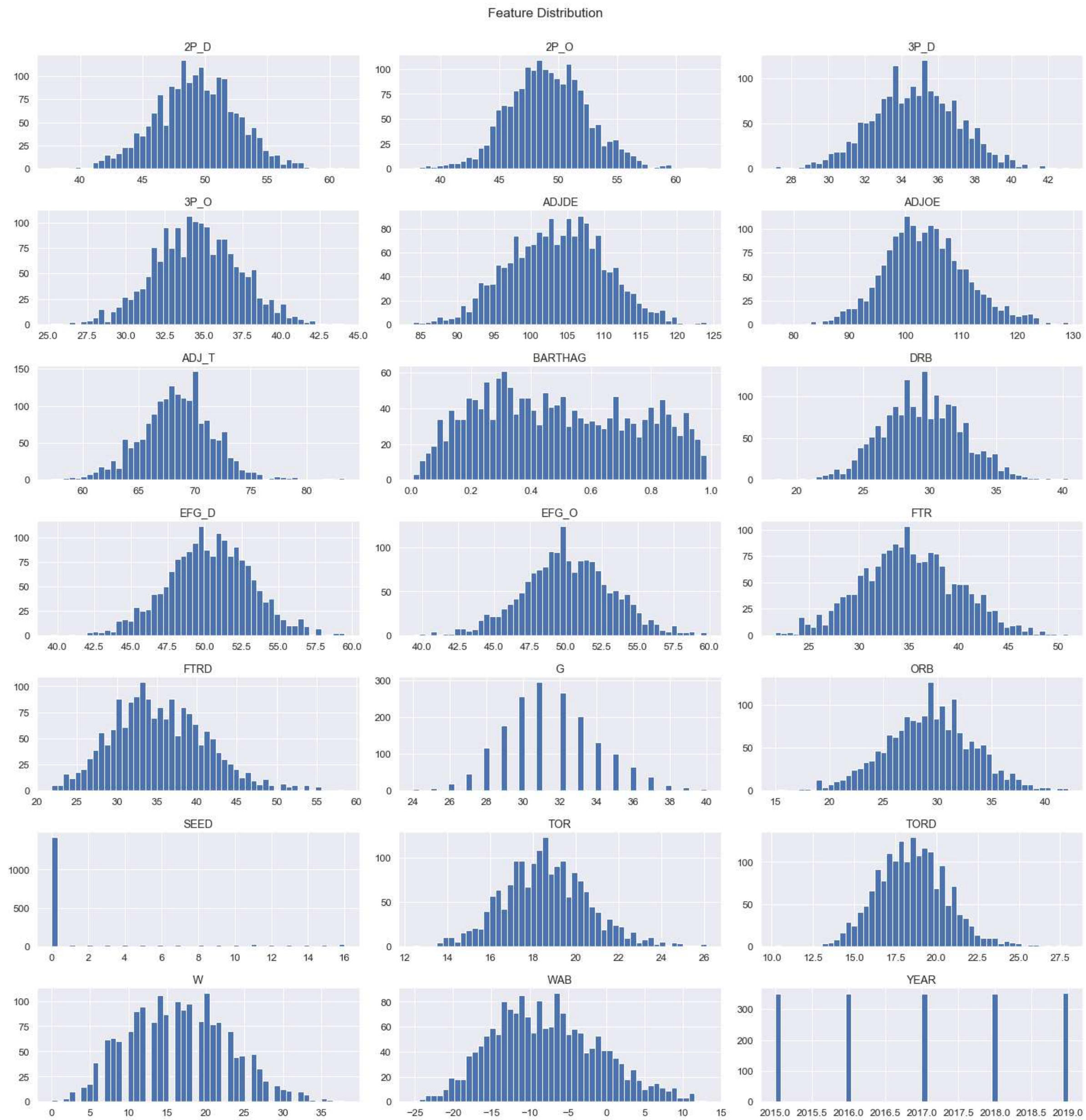
Post season results = Only 10 teams made it to Final Four in last 5 years. N means teams got eliminated before R68.

In [10]:
```
df['POSTSEASON'].value_counts()
```
Out[10]:
```
N            1417
R64           160
R32            80
S16            40
E8             20
R68            20
F4             10
Champions       5
2ND             5
Name: POSTSEASON, dtype: int64
```

Data Visualization

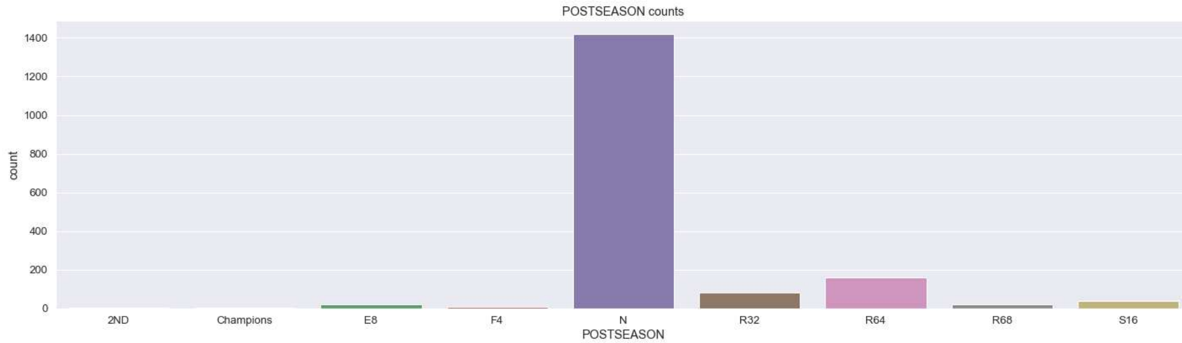
The dataset is Normally distributed.

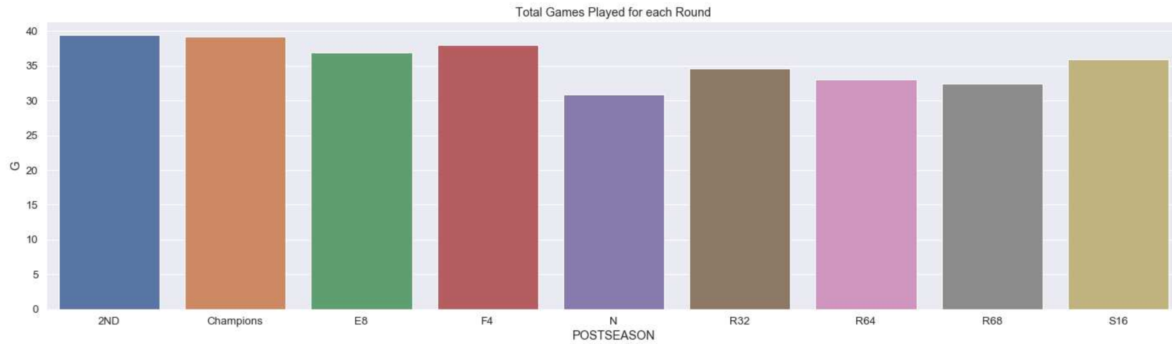
In [11]:
```
fig, ax = plt.subplots(nrows=7, ncols=3, sharex=False, sharey=False, figsize=(20,2
0))

df.hist(bins=50, ax = ax)

plt.suptitle('Feature Distribution', x=0.5, y=1.02, ha='center', fontsize='large')

plt.tight_layout()

plt.show();
```

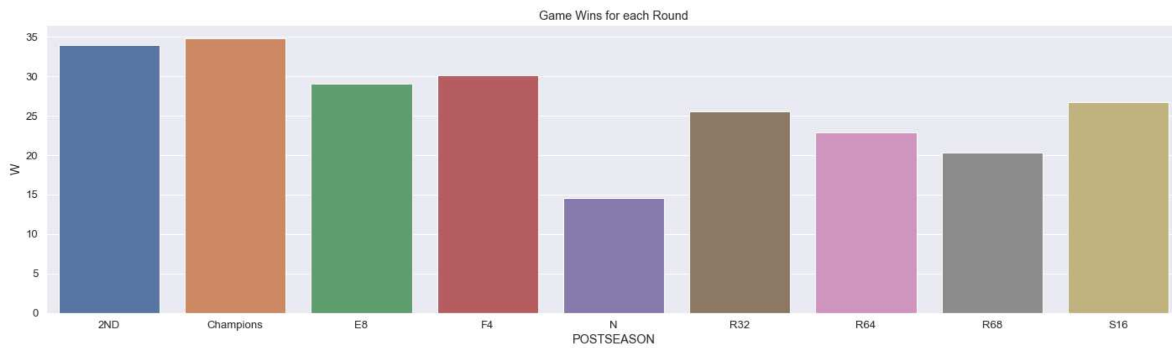


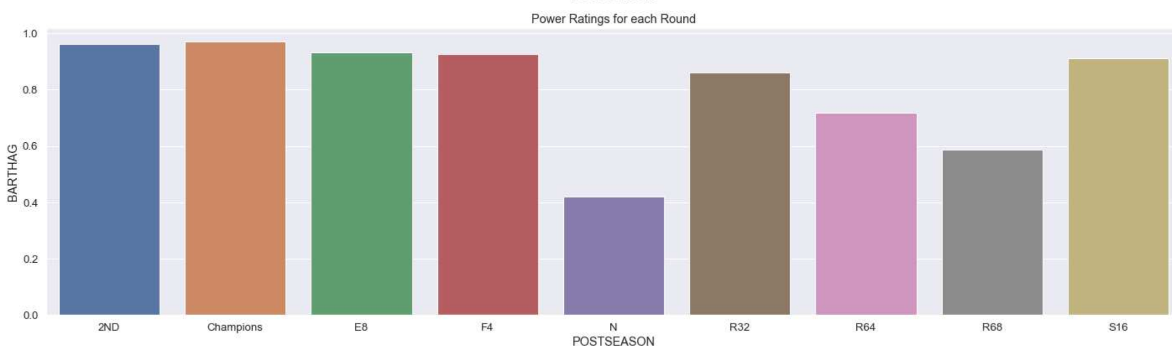
Below are each visuals to see the data:

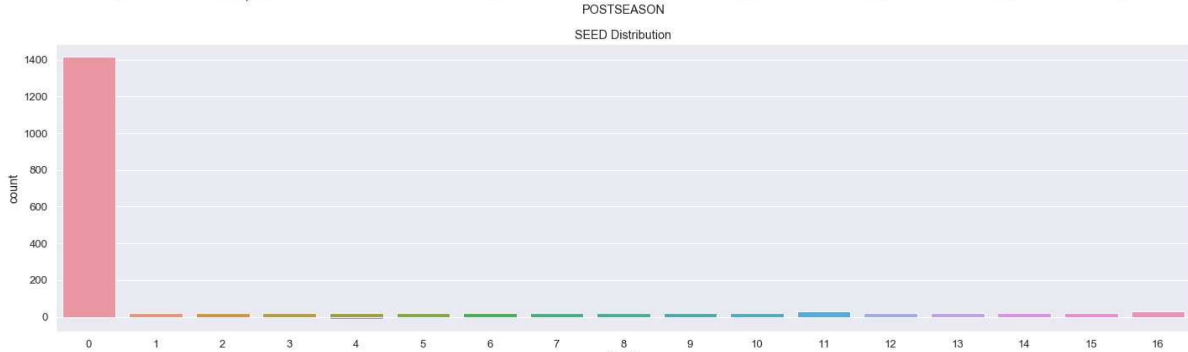
In [12]:
```
fig = plt.figure(figsize=(20,40))

plt.subplot(7,1,1)
plt.title("POSTSEASON counts")
sns.countplot(df.POSTSEASON)

plt.subplot(7,1,2)
plt.title("Total Games Played for each Round")
sns.barplot(x=df.POSTSEASON, y=df.G, ci=None)

plt.subplot(7,1,3)
plt.title("Game Wins for each Round")
sns.barplot(x=df.POSTSEASON, y=df.W, ci=None)

plt.subplot(7,1,4)
plt.title("Power Ratings for each Round")
sns.barplot(x=df.POSTSEASON, y=df.BARTHAG, ci=None)

plt.subplot(7,1,5)
plt.title("Wins Above Bubble for each Round")
sns.barplot(x=df.POSTSEASON, y=df.WAB, ci=None)

plt.subplot(7,1,5)
plt.title("SEED Distribution")
sns.countplot(df.SEED)


plt.tight_layout()
plt.show()
```









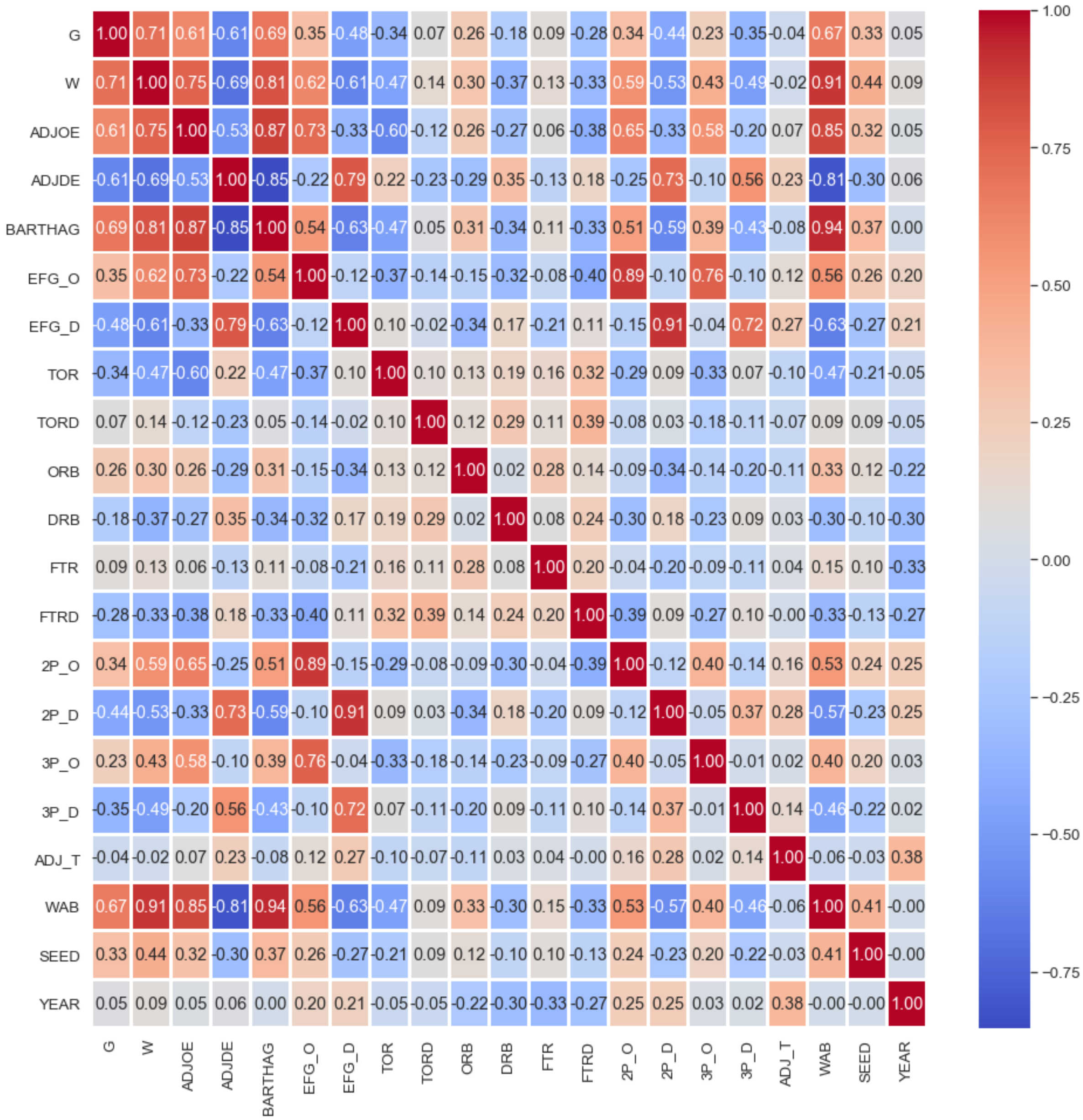


Now we check any correlation between features:

In [13]:
```
df.corr()
```
Out[13]:

	G	W	ADJOE	ADJDE	BARTHAG	EFG_O	EFG_D	TOR	TORD
G	1.000000	0.708838	0.613432	-0.606166	0.688059	0.346425	-0.481942	-0.336407	0.065020
W	0.708838	1.000000	0.754532	-0.690753	0.814512	0.617839	-0.609144	-0.467073	0.138805
ADJOE	0.613432	0.754532	1.000000	-0.528576	0.870686	0.732683	-0.333693	-0.601024	-0.116231
ADJDE	-0.606166	-0.690753	-0.528576	1.000000	-0.852432	-0.221381	0.792320	0.219779	-0.234615
BARTHAG	0.688059	0.814512	0.870686	-0.852432	1.000000	0.543153	-0.627696	-0.472329	0.054377
EFG_O	0.346425	0.617839	0.732683	-0.221381	0.543153	1.000000	-0.120335	-0.367975	-0.144287
EFG_D	-0.481942	-0.609144	-0.333693	0.792320	-0.627696	-0.120335	1.000000	0.101070	-0.020831
TOR	-0.336407	-0.467073	-0.601024	0.219779	-0.472329	-0.367975	0.101070	1.000000	0.103437
TORD	0.065020	0.138805	-0.116231	-0.234615	0.054377	-0.144287	-0.020831	0.103437	1.000000
ORB	0.261046	0.296395	0.261351	-0.294066	0.310917	-0.147990	-0.341636	0.134433	0.118496
DRB	-0.184134	-0.366715	-0.266665	0.347646	-0.337804	-0.319901	0.172261	0.188585	0.289078
FTR	0.090549	0.126931	0.063637	-0.125265	0.112072	-0.083514	-0.205807	0.161369	0.111844
FTRD	-0.279593	-0.329245	-0.382290	0.180823	-0.327932	-0.404445	0.107161	0.316898	0.392412
2P_O	0.339290	0.585806	0.646011	-0.251354	0.512045	0.893530	-0.148439	-0.288945	-0.079076
2P_D	-0.439340	-0.529558	-0.328822	0.728488	-0.588121	-0.104079	0.907933	0.091686	0.027203
3P_O	0.225821	0.432743	0.579193	-0.102715	0.386597	0.763028	-0.043770	-0.333397	-0.177276
3P_D	-0.349726	-0.485485	-0.198275	0.564135	-0.427750	-0.100545	0.722404	0.066363	-0.106798
ADJ_T	-0.040433	-0.016057	0.070476	0.227852	-0.079611	0.120142	0.273412	-0.102687	-0.065216
WAB	0.666595	0.905029	0.851663	-0.809486	0.941776	0.562904	-0.629864	-0.470286	0.094765
SEED	0.331670	0.439614	0.321272	-0.299561	0.369614	0.263457	-0.268033	-0.205446	0.086535
YEAR	0.052233	0.091829	0.048861	0.055367	0.000228	0.196195	0.211657	-0.054840	-0.051754

In [14]:
```
plt.figure(figsize=(16,16))
sns.heatmap(df.corr(),cmap="coolwarm",annot=True,fmt='.2f',linewidths=2)
plt.show()
```



WAB(Wins Above Bubble) is highly correlated with W, ADJOE and BARTHAG. It also highly negative correlation with ADJDE. Means Offensive Teams are more likely to succeed in each rounds.

Data Preprocessing is next by checking missing values, preparing the data for modeling

Treat Missing Values

In [15]:
```
df.isnull().sum()
```

Out[15]:
```
TEAM          0
CONF          0
G             0
W             0
ADJOE         0
ADJDE         0
BARTHAG       0
EFG_O         0
EFG_D         0
TOR           0
TORD          0
ORB           0
DRB           0
FTR           0
FTRD          0
2P_O          0
2P_D          0
3P_O          0
3P_D          0
ADJ_T         0
WAB           0
POSTSEASON    0
SEED          0
YEAR          0
dtype: int64
```

Treat Duplicate Values

In [16]:
```
df.duplicated(keep='first').sum()
```

Out[16]:
```
0
```

Drop unwanted features

In [17]:
```
df.columns
```

Out[17]:
```
Index(['TEAM', 'CONF', 'G', 'W', 'ADJOE', 'ADJDE', 'BARTHAG', 'EFG_O', 'EFG_D',
       'TOR', 'TORD', 'ORB', 'DRB', 'FTR', 'FTRD', '2P_O', '2P_D', '3P_O',
       '3P_D', 'ADJ_T', 'WAB', 'POSTSEASON', 'SEED', 'YEAR'],
      dtype='object')
```

In [18]:
```
df.drop(['TEAM', 'CONF', 'SEED', 'YEAR'],axis=1,inplace=True)
```

```
In [19]: df
```

Out[19]:

	G	W	ADJOE	ADJDE	BARTHAG	EFG_O	EFG_D	TOR	TORD	ORB	DRB	FTR	FTRD	2P_O	2P_D
0	40	33	123.3	94.9	0.9531	52.6	48.1	15.4	18.2	40.7	30.0	32.3	30.4	53.9	44.6
1	40	35	123.1	90.9	0.9703	56.1	46.7	16.3	20.6	28.2	29.4	34.1	30.0	57.4	44.1
2	36	24	118.3	103.3	0.8269	54.0	49.5	15.3	14.8	32.7	32.1	32.9	26.0	52.9	46.5
3	37	29	119.9	91.0	0.9600	54.8	48.4	15.1	18.8	29.9	25.2	32.1	33.4	52.6	46.3
4	37	32	120.9	90.4	0.9662	55.7	45.1	17.8	18.5	32.2	27.9	38.6	37.3	52.7	43.4
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1752	36	22	111.8	96.6	0.8425	49.6	48.5	17.6	17.9	33.8	28.6	35.7	32.3	47.4	45.4
1753	34	25	114.9	88.7	0.9513	55.2	43.0	18.2	18.3	31.3	28.4	43.4	34.3	52.3	41.4
1754	35	25	110.3	93.3	0.8733	46.1	52.7	18.7	28.0	40.1	31.1	40.4	55.5	45.5	51.8
1755	34	29	114.3	91.5	0.9277	50.3	45.8	15.0	21.3	34.5	27.4	36.2	36.6	48.9	42.6
1756	37	23	115.7	95.1	0.9049	53.3	50.0	18.1	18.8	31.3	27.3	38.5	33.3	53.7	48.9

1757 rows × 20 columns

We dropped 4 features because we are using numerical features to predict who will be in Final Four.

Narrow down for teams qualified from Round 68 onwards

```
In [20]: df["POSTSEASON"].value_counts()
```

```
Out[20]: N            1417
         R64           160
         R32            80
         S16            40
         E8             20
         R68            20
         F4             10
         Champions       5
         2ND             5
         Name: POSTSEASON, dtype: int64
```

```
In [21]: df1 = df[df['POSTSEASON'].str.contains('Champions|2ND|F4|S16|E8|R32|R64|R68', na=False)]
```

In [22]:
```
df1
```

Out[22]:

	G	W	ADJOE	ADJDE	BARTHAG	EFG_O	EFG_D	TOR	TORD	ORB	DRB	FTR	FTRD	2P_O	2P_D
0	40	33	123.3	94.9	0.9531	52.6	48.1	15.4	18.2	40.7	30.0	32.3	30.4	53.9	44.6
1	40	35	123.1	90.9	0.9703	56.1	46.7	16.3	20.6	28.2	29.4	34.1	30.0	57.4	44.1
2	36	24	118.3	103.3	0.8269	54.0	49.5	15.3	14.8	32.7	32.1	32.9	26.0	52.9	46.5
3	37	29	119.9	91.0	0.9600	54.8	48.4	15.1	18.8	29.9	25.2	32.1	33.4	52.6	46.3
4	37	32	120.9	90.4	0.9662	55.7	45.1	17.8	18.5	32.2	27.9	38.6	37.3	52.7	43.4
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1752	36	22	111.8	96.6	0.8425	49.6	48.5	17.6	17.9	33.8	28.6	35.7	32.3	47.4	45.4
1753	34	25	114.9	88.7	0.9513	55.2	43.0	18.2	18.3	31.3	28.4	43.4	34.3	52.3	41.4
1754	35	25	110.3	93.3	0.8733	46.1	52.7	18.7	28.0	40.1	31.1	40.4	55.5	45.5	51.8
1755	34	29	114.3	91.5	0.9277	50.3	45.8	15.0	21.3	34.5	27.4	36.2	36.6	48.9	42.6
1756	37	23	115.7	95.1	0.9049	53.3	50.0	18.1	18.8	31.3	27.3	38.5	33.3	53.7	48.9

340 rows × 20 columns

In [23]:
```
df1.reset_index(drop=True, inplace=True)
```

In [24]:
```
df1
```

Out[24]:

	G	W	ADJOE	ADJDE	BARTHAG	EFG_O	EFG_D	TOR	TORD	ORB	DRB	FTR	FTRD	2P_O	2P_D
0	40	33	123.3	94.9	0.9531	52.6	48.1	15.4	18.2	40.7	30.0	32.3	30.4	53.9	44.6
1	40	35	123.1	90.9	0.9703	56.1	46.7	16.3	20.6	28.2	29.4	34.1	30.0	57.4	44.1
2	36	24	118.3	103.3	0.8269	54.0	49.5	15.3	14.8	32.7	32.1	32.9	26.0	52.9	46.5
3	37	29	119.9	91.0	0.9600	54.8	48.4	15.1	18.8	29.9	25.2	32.1	33.4	52.6	46.3
4	37	32	120.9	90.4	0.9662	55.7	45.1	17.8	18.5	32.2	27.9	38.6	37.3	52.7	43.4
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
335	36	22	111.8	96.6	0.8425	49.6	48.5	17.6	17.9	33.8	28.6	35.7	32.3	47.4	45.4
336	34	25	114.9	88.7	0.9513	55.2	43.0	18.2	18.3	31.3	28.4	43.4	34.3	52.3	41.4
337	35	25	110.3	93.3	0.8733	46.1	52.7	18.7	28.0	40.1	31.1	40.4	55.5	45.5	51.8
338	34	29	114.3	91.5	0.9277	50.3	45.8	15.0	21.3	34.5	27.4	36.2	36.6	48.9	42.6
339	37	23	115.7	95.1	0.9049	53.3	50.0	18.1	18.8	31.3	27.3	38.5	33.3	53.7	48.9

340 rows × 20 columns

Now we need to encode 1 for teams made to Final Four to Champion and the rest are 0.

Focusing on teams winning F4 to Champions

In [25]:
```
df1["POSTSEASON"] = df1["POSTSEASON"].replace(to_replace=["2ND","Champions","F4"],
value=1)
```

In [26]:
```
df1["POSTSEASON"] = df1["POSTSEASON"].replace(to_replace=["E8","S16","R32","R64","R68"], value=0)
```

In [27]:
```
df1
```

Out[27]:

	G	W	ADJOE	ADJDE	BARTHAG	EFG_O	EFG_D	TOR	TORD	ORB	DRB	FTR	FTRD	2P_O	2P_D
0	40	33	123.3	94.9	0.9531	52.6	48.1	15.4	18.2	40.7	30.0	32.3	30.4	53.9	44.6
1	40	35	123.1	90.9	0.9703	56.1	46.7	16.3	20.6	28.2	29.4	34.1	30.0	57.4	44.1
2	36	24	118.3	103.3	0.8269	54.0	49.5	15.3	14.8	32.7	32.1	32.9	26.0	52.9	46.5
3	37	29	119.9	91.0	0.9600	54.8	48.4	15.1	18.8	29.9	25.2	32.1	33.4	52.6	46.3
4	37	32	120.9	90.4	0.9662	55.7	45.1	17.8	18.5	32.2	27.9	38.6	37.3	52.7	43.4
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
335	36	22	111.8	96.6	0.8425	49.6	48.5	17.6	17.9	33.8	28.6	35.7	32.3	47.4	45.4
336	34	25	114.9	88.7	0.9513	55.2	43.0	18.2	18.3	31.3	28.4	43.4	34.3	52.3	41.4
337	35	25	110.3	93.3	0.8733	46.1	52.7	18.7	28.0	40.1	31.1	40.4	55.5	45.5	51.8
338	34	29	114.3	91.5	0.9277	50.3	45.8	15.0	21.3	34.5	27.4	36.2	36.6	48.9	42.6
339	37	23	115.7	95.1	0.9049	53.3	50.0	18.1	18.8	31.3	27.3	38.5	33.3	53.7	48.9

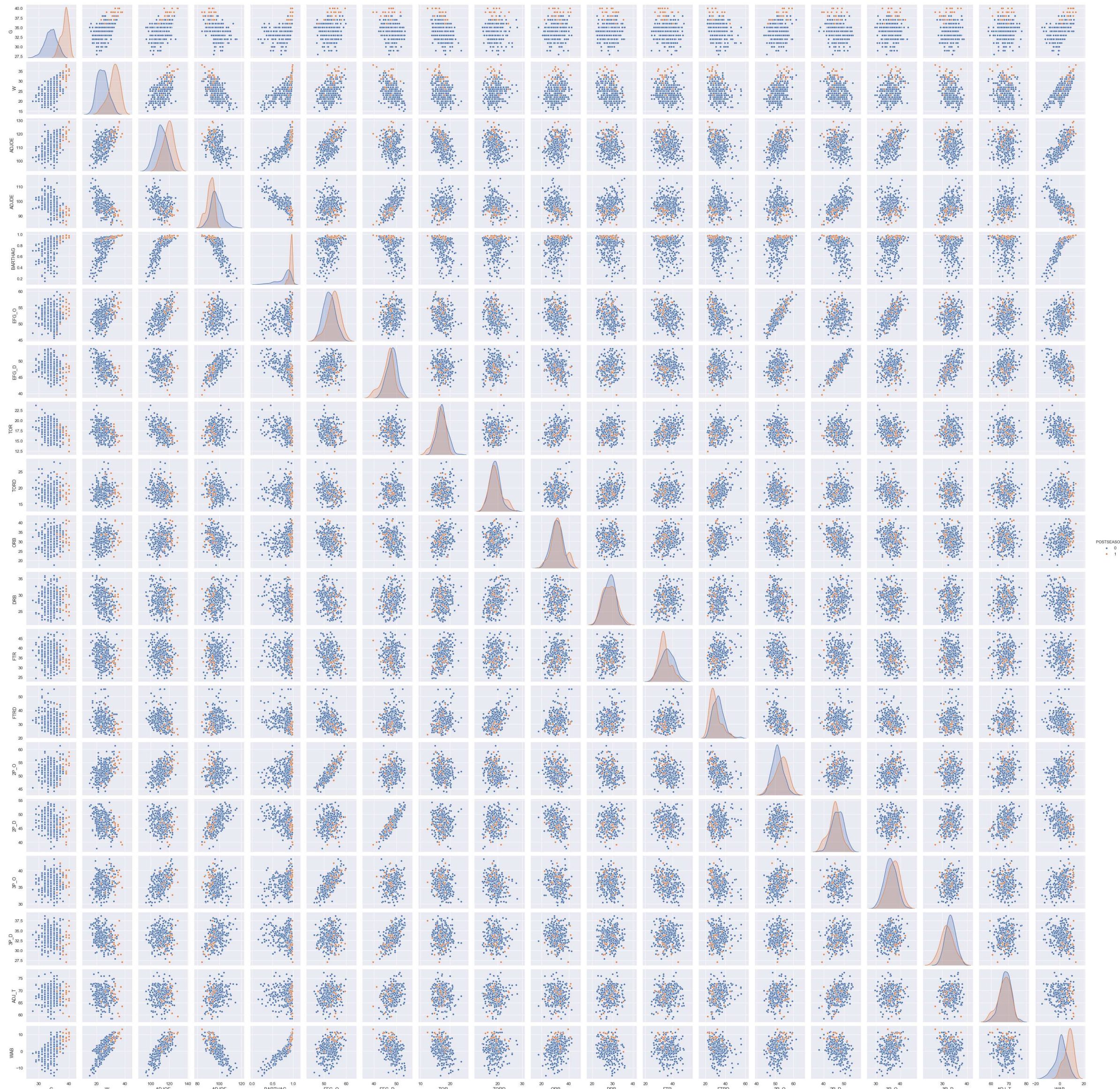
340 rows × 20 columns

In [28]:
```
df1["POSTSEASON"].value_counts()
```

Out[28]:
```
0    320
1     20
Name: POSTSEASON, dtype: int64
```

The number of teams as expected are few to made it at least Final Four.

In [29]:
```
sns.pairplot(df1, hue="POSTSEASON");
```



Blue color is 0, orange is 1

Create and save processed dataset

In [30]:
```
df1.to_csv("train.csv",index=False)
```

In [31]:
```
df1 = pd.read_csv("train.csv")
```

In [32]:
```
df1
```

Out[32]:

	G	W	ADJOE	ADJDE	BARTHAG	EFG_O	EFG_D	TOR	TORD	ORB	DRB	FTR	FTRD	2P_O	2P_D
0	40	33	123.3	94.9	0.9531	52.6	48.1	15.4	18.2	40.7	30.0	32.3	30.4	53.9	44.6
1	40	35	123.1	90.9	0.9703	56.1	46.7	16.3	20.6	28.2	29.4	34.1	30.0	57.4	44.1
2	36	24	118.3	103.3	0.8269	54.0	49.5	15.3	14.8	32.7	32.1	32.9	26.0	52.9	46.5
3	37	29	119.9	91.0	0.9600	54.8	48.4	15.1	18.8	29.9	25.2	32.1	33.4	52.6	46.3
4	37	32	120.9	90.4	0.9662	55.7	45.1	17.8	18.5	32.2	27.9	38.6	37.3	52.7	43.4
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
335	36	22	111.8	96.6	0.8425	49.6	48.5	17.6	17.9	33.8	28.6	35.7	32.3	47.4	45.4
336	34	25	114.9	88.7	0.9513	55.2	43.0	18.2	18.3	31.3	28.4	43.4	34.3	52.3	41.4
337	35	25	110.3	93.3	0.8733	46.1	52.7	18.7	28.0	40.1	31.1	40.4	55.5	45.5	51.8
338	34	29	114.3	91.5	0.9277	50.3	45.8	15.0	21.3	34.5	27.4	36.2	36.6	48.9	42.6
339	37	23	115.7	95.1	0.9049	53.3	50.0	18.1	18.8	31.3	27.3	38.5	33.3	53.7	48.9

340 rows × 20 columns

Formulating at least 3 hypothesis about this data

In [33]:
```
df1.groupby(["POSTSEASON"]).mean()
```

Out[33]:

	G	W	ADJOE	ADJDE	BARTHAG	EFG_O	EFG_D	TOR	TORD
POSTSEASON									
0	34.00625	24.2375	111.333437	97.315	0.783932	52.35125	48.114688	17.405313	18.865938
1	38.65000	32.2500	119.375000	91.625	0.946165	54.35500	46.255000	16.545000	19.175000

In [34]:
```
df1.groupby(["POSTSEASON"])['W'].mean()
```

Out[34]:
```
POSTSEASON
0    24.2375
1    32.2500
Name: W, dtype: float64
```

In [35]:
```
df1.groupby(["POSTSEASON"])['WAB'].mean()
```

Out[35]:
```
POSTSEASON
0    1.040937
1    7.485000
Name: WAB, dtype: float64
```

In [36]:
```
df1.groupby(["POSTSEASON"])['BARTHAG'].mean()
```

Out[36]:
```
POSTSEASON
0    0.783932
1    0.946165
Name: BARTHAG, dtype: float64
```

Hypothesis 1: Mean Winning games (W) is at least 32 Wins. We need to test is 32 wins is normal (H0) or by chance? (H1)

Hypothesis 2: Mean Wins Above Bubble (WAB) is at least 7. We need to test if 7 is true (H0) or false? (H1)

Hypothesis 3: Mean Power Rating (BARTHAG) is 0.946. We need to test minimum 0.9 (H0) or by less (H1)

Conducting a formal significance test for one of the hypotheses and discuss the results

In [37]:
```
a = df1["POSTSEASON"]
b = df1["W"]
```

In [38]:
```
correlation , pvalue = pearsonr(a,b)
```

In [39]:
```
print("Correlation is %.3f", correlation)
```

```
Correlation is %.3f 0.44626165854156674
```

In [40]:
```
print("P value is %.4f", pvalue)
```

```
P value is %.4f 4.816984468590114e-18
```

The significance level is the probability of rejecting the null hypothesis when it is true. For example, a significance level of 0.05 indicates a 5% risk of concluding that a difference exists when there is no actual difference. Lower significance levels indicate that you require stronger evidence before you will reject the null hypothesis.

In [41]:
```
alpha = 0.05  # Set p-value
```

In [42]:
```
if pvalue < alpha:
    print("Significant correlation")
else:
    print("No significant correlation")
```

```
Significant correlation
```

The result for pvalue is much smaller than 0.05, hence it is statistically significant and the wins are not by chance. Hence **Null Hypothesis (H0) is not rejected**.

The result for pvalue is much smaller than 0.05, hence it is statistically significant and the wins are not by chance. Hence **Null Hypothesis (H0) is not rejected**.

Suggestions for next steps in analyzing this data

There are other features can be looked into like Effective Field Goal Percentage, Turnover Percentage, Offensive/Defensive Rebound Percentage, Free Throw Rate and Adjusted Tempo which are not covered in this report. Detailed analysis on these attributes may give data insights on teams performance to make it to Final Four.

A paragraph that summarizes the quality of this data set and a request for additional data if needed

The dataset can only give us acceptable results if used for prediction. We suggest more years of data (10 years or more) so that data can be further analysed and model prediction will be much better.