In [3]:
```
from NBA_Roster_Analysis import  NBA_Roster_Analysis
```

In [4]:
```
help(NBA_Roster_Analysis)
```

```
Help on class NBA_Roster_Analysis in module NBA_Roster_Analysis:

class NBA_Roster_Analysis(builtins.object)
 |  Methods defined here:
 |
 |  __init__(self)
 |      initializing module, importing and defining data required for ana
lysis
 |
 |  adjust_to_minutes(self, minutes: float, player_stats: pandas.core.fra
me.DataFrame)
 |      given minutes and players statistics, this function adjust the pl
ayers stats
 |      Args:
 |          minutes (float): minutes played
 |          player_stats (pd.Dataframe): a dataframe containing the stats
of the player
 |
 |      Returns:
 |          adjusted_player_stats (list)
 |
 |  download_player_data(self, years, filename, filetype='csv')
 |      Downloading player data for years specified
 |
 |  download_team_data(self, years, filename, filetype='csv')
 |      Downloading player data for years specified
 |
 |  k_nearest_neighbors(self, team: list, minutes_selection_method: str =
'sample', stats_selection_method: str = 'prime', prime_window=None, k=5,
visualize=False)
 |      given a list of 8 players, how their minutes and stats, this func
tion will output K nearest historical teams, as well as option to view lo
cation
 |      Args:
 |          team (list): name of NBA player
 |          minutes_selection_method (str): minutes the player will play
 |          stats_selection_method (str): choose from 'best' or'prime'.
'best' will sample stats from player's best season, 'prime' will sample s
tats from a window of seasons defined by prime_window
 |          prime_window (int): number of years to consider a player's 'p
rime', the default is 5 years chosen when stats_selection_method='prime'
 |          output_df (bool):
 |      Returns:
 |          visuals and k hisrorical nba teams
 |
 |  normalize_sampled_stats(self, team: list, minutes_selection_method: s
tr = 'sample', stats_selection_method: str = 'prime', prime_window=None)
 |      given a list of 8 players, how their minutes and stats, this func
tion will give their normalized stats
 |      Args:
 |          team (list): name of NBA player
 |          minutes_selection_method (str): minutes the player will play
 |          stats_selection_method (str): choose from 'best' or'prime'.
'best' will sample stats from player's best season, 'prime' will sample s
tats from a window of seasons defined by prime_window
```

```
 |          prime_window (int): number of years to consider a player's 'p
rime', the default is 5 years chosen when stats_selection_method='prime'
 |          output_df (bool):
 |      Returns:
 |          normalized stats
 |
 |  normalize_team_stats(self, team_stats)
 |      normalizing team statistics with MinMaxScaler
 |
 |      Args:
 |          team_stats (pd.DataFrame): team stats
 |
 |      Returns:
 |          normalized_team_stats (pd.DataFrame): normalized team stats
 |
 |  player_stats_sampling(self, player: str, minutes: float, stats_select
ion_method: str = 'prime', prime_window=None)
 |      given a player, and minutes he will play, this function will retu
rn a sample of his statistics
 |      Args:
 |          player (string): name of NBA player
 |          minutes(float): minutes the player will play
 |          stats_selection_method (str): choose from 'best' or'prime'.
'best' will sample stats from player's best season, 'prime' will sample s
tats from a window of seasons defined by prime_window
 |          prime_window (int): number of years to consider a player's 'p
rime', the default is 5 years chosen when stats_selection_method='prime'
 |      Returns:
 |          adjusted_player_stats (list)
 |
 |      Raise:
 |          Exception: when player chosen does not exist
 |
 |  predict_roster_cluster(self, team: list, minutes_selection_method: st
r = 'sample', stats_selection_method: str = 'prime', prime_window=None)
 |      given a list of 8 players, how their minutes and stats, this func
tion will predict the cluster it belongs to
 |      Args:
 |          team (list): name of NBA player
 |          minutes_selection_method (str): minutes the player will play
 |          stats_selection_method (str): choose from 'best' or'prime'.
'best' will sample stats from player's best season, 'prime' will sample s
tats from a window of seasons defined by prime_window
 |          prime_window (int): number of years to consider a player's 'p
rime', the default is 5 years chosen when stats_selection_method='prime'
 |          output_df (bool):
 |      Returns:
 |          cluster
 |
 |  predict_team_stats(self, team: list, minutes_selection_method: str =
'sample', stats_selection_method: str = 'prime', prime_window=None)
 |      given a list of 8 players, how their minutes and stats, this func
tion will predict their team stats from neural networks trained
 |      Args:
 |          team (list): name of NBA player
 |          minutes_selection_method (str): minutes the player will play
 |          stats_selection_method (str): choose from 'best' or'prime'.
```

```
'best' will sample stats from player's best season, 'prime' will sample s
tats from a window of seasons defined by prime_window
 |          prime_window (int): number of years to consider a player's 'p
rime', the default is 5 years chosen when stats_selection_method='prime'
 |          output_df (bool):
 |      Returns:
 |          predicted team stats
 |
 |  team_stats_sampling(self, team: list, minutes_selection_method: str =
'sample', stats_selection_method: str = 'prime', prime_window=None, outpu
t_df=False)
 |      given a list of 8 players, how their minutes and stats, this func
tion will sample their player stats
 |      Args:
 |          team (list): name of NBA player
 |          minutes_selection_method (str): minutes the player will play
 |          stats_selection_method (str): choose from 'best' or'prime'.
'best' will sample stats from player's best season, 'prime' will sample s
tats from a window of seasons defined by prime_window
 |          prime_window (int): number of years to consider a player's 'p
rime', the default is 5 years chosen when stats_selection_method='prime'
 |          output_df (bool):
 |      Returns:
 |          team_stats (np.array)
 |          or
 |          team_stats (pd.DataFrame)
 |
 |      Raise:
 |          Exception: when player chosen does not exist
 |
 |  view_available_players(self)
 |      viewing list of available players to choose roster for
 |
 |      Args:
 |          None
 |
 |      Returns:
 |          player (list): list of unique players
 |
 |  visualize_new_roster(self, normalized_stats, predicted_stats, cluste
r)
 |      shows visual location of new roster
 |
 |  visualize_teams(self, color: str = 'Clusters', hover_data: list = ['W
IN%'])
 |      visualizing NBA team through 1999 - 2020 in 2D w/ Principal Compo
nent Analysis
 |
 |      Args:
 |          color (str): team statistic to color data points by
 |          hover_data (list): list of team statistics to show when hover
ing on datapoints
 |
 |      Returns:
 |          fig: visual of teams with Plotly
 |          df (pd.Dataframe): dataframe of teams
 |
```

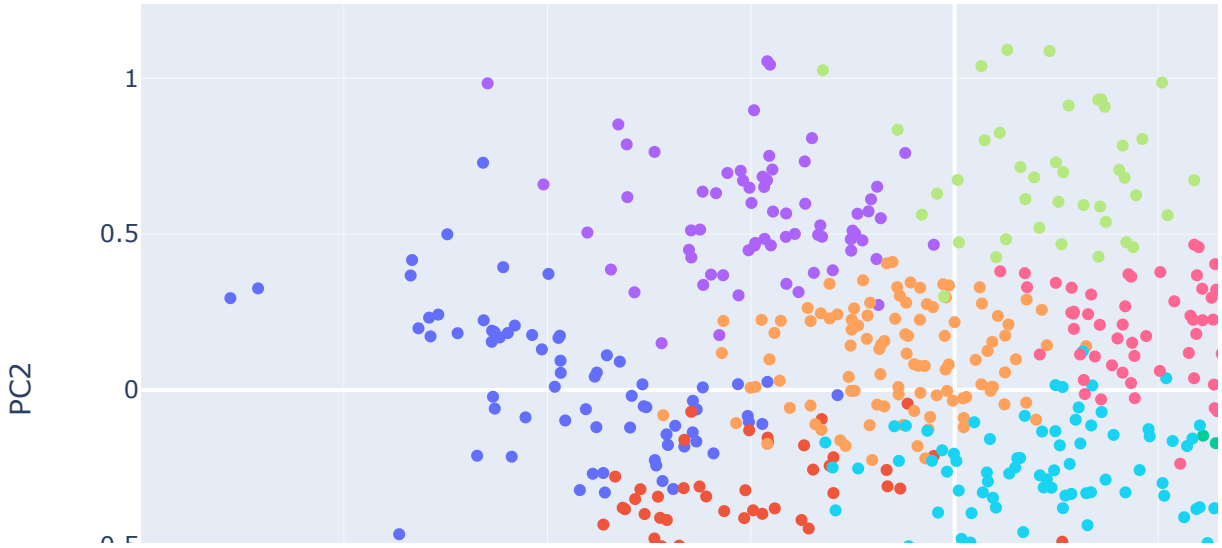
```
 |  ----------------------------------------------------------------------
-
 |  Data descriptors defined here:
 |
 |  __dict__
 |      dictionary for instance variables (if defined)
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
```

Initializing Module

In [5]:
```
nba_analysis=NBA_Roster_Analysis()
```

Viewing Teams

In [6]:
```
nba_analysis.visualize_teams(hover_data=['WIN%','TEAM'])
```



Out[6]:

	TEAM	Clusters	WIN%	PTS	FGM	FGA	FG%	3PM	3PA	3P%	...	AST\nRATIO	OR
0	Atlanta Hawks 1999-00	2	0.341	94.3	36.6	83.0	44.1	3.1	9.9	31.7	...	14.6	
1	Boston Celtics 1999-00	2	0.427	99.3	37.2	83.9	44.4	5.1	15.4	33.1	...	15.9	
2	Charlotte Hornets 1999-00	5	0.598	98.4	35.8	79.7	44.9	4.1	12.2	33.9	...	18.4	
3	Chicago Bulls 1999-00	2	0.207	84.8	31.3	75.4	41.5	4.1	12.6	32.9	...	15.8	
4	Cleveland Cavaliers 1999-00	2	0.390	97.0	36.3	82.1	44.2	4.2	11.2	37.3	...	17.3	
...	...	...	...	...	...	...	...	...	...	...	...	...	

	TEAM	Clusters	WIN%	PTS	FGM	FGA	FG%	3PM	3PA	3P%	...	AST\nRATIO	OR
620	Sacramento Kings 2019-20	3	0.431	110.1	40.9	88.4	46.2	12.7	34.9	36.4	...	17.4	
621	San Antonio Spurs 2019-20	3	0.451	114.1	42.2	89.4	47.2	10.7	28.5	37.6	...	17.9	
622	Toronto Raptors 2019-20	6	0.736	112.8	40.2	87.9	45.8	13.8	37.0	37.4	...	18.1	
623	Utah Jazz 2019-20	6	0.611	111.3	40.1	85.1	47.1	13.4	35.2	38.0	...	16.8	
624	Washington Wizards 2019-20	4	0.347	114.4	41.5	90.9	45.7	12.0	32.6	36.8	...	17.6	

625 rows × 39 columns

Hypothetical Roster

In [7]:
```
team=['LeBron James','Kevin Durant',
      'Stephen Curry',"Shaquille O'Neal",
      'Carmelo Anthony','Kevin Garnett',
      'Paul Pierce','Ray Allen']
```

In [8]:
```
nba_analysis.team_stats_sampling(team,minutes_selection_method='sample',sta
```

Out[8]:

	PLAYER	POS	MIN	PTS	FGM	FGA	FG%	3PM	3PA	3
0	LeBron James	SF	35.881300	26.109912	9.454136	17.956513	52.833333	1.237286	3.299430	37.06
1	Kevin Durant	PF	32.874304	25.367807	8.396596	16.555832	50.766667	1.928547	4.776862	40.33
2	Stephen Curry	PG	33.486020	27.188346	9.074068	18.249711	49.533333	4.367742	9.852813	44.00
3	Shaquille O'Neal	C	28.994951	21.470309	8.452680	14.698132	57.500000	0.000000	0.000000	0.00
4	Carmelo Anthony	PF	31.773532	22.773925	7.986789	17.970276	44.366667	1.649446	4.427459	37.20
5	Kevin Garnett	PF	22.969325	13.457972	5.189805	10.241477	50.900000	0.059199	0.236797	25.43
6	Paul Pierce	SF	19.137814	12.726243	4.150564	9.399332	44.300000	0.888253	2.454808	35.33
7	Ray Allen	SG	18.097128	10.758344	3.731849	8.338107	44.933333	1.217926	3.060429	40.03

8 rows × 38 columns

In [9]:
```
nba_analysis.predict_team_stats(team,minutes_selection_method='sample',stat
```

```
WARNING:tensorflow:5 out of the last 5 calls to <function Model.make_pred
ict_function.<locals>.predict_function at 0x7f8b0b61d670> triggered tf.fu
nction retracing. Tracing is expensive and the excessive number of tracin
gs could be due to (1) creating @tf.function repeatedly in a loop, (2) pa
ssing tensors with different shapes, (3) passing Python objects instead o
f tensors. For (1), please define your @tf.function outside of the loop.
For (2), @tf.function has experimental_relax_shapes=True option that rela
xes argument shapes that can avoid unnecessary retracing. For (3), please
refer to https://www.tensorflow.org/guide/function#controlling_retracing
 (https://www.tensorflow.org/guide/function#controlling_retracing) and ht
tps://www.tensorflow.org/api_docs/python/tf/function (https://www.tensorf
low.org/api_docs/python/tf/function) for  more details.
WARNING:tensorflow:6 out of the last 6 calls to <function Model.make_pred
ict_function.<locals>.predict_function at 0x7f8b0b2561f0> triggered tf.fu
nction retracing. Tracing is expensive and the excessive number of tracin
gs could be due to (1) creating @tf.function repeatedly in a loop, (2) pa
ssing tensors with different shapes, (3) passing Python objects instead o
f tensors. For (1), please define your @tf.function outside of the loop.
 For (2), @tf.function has experimental_relax_shapes=True option that rel
axes argument shapes that can avoid unnecessary retracing. For (3), pleas
e refer to https://www.tensorflow.org/guide/function#controlling_retracin
g (https://www.tensorflow.org/guide/function#controlling_retracing) and h
ttps://www.tensorflow.org/api_docs/python/tf/function (https://www.tensor
flow.org/api_docs/python/tf/function) for  more details.
```

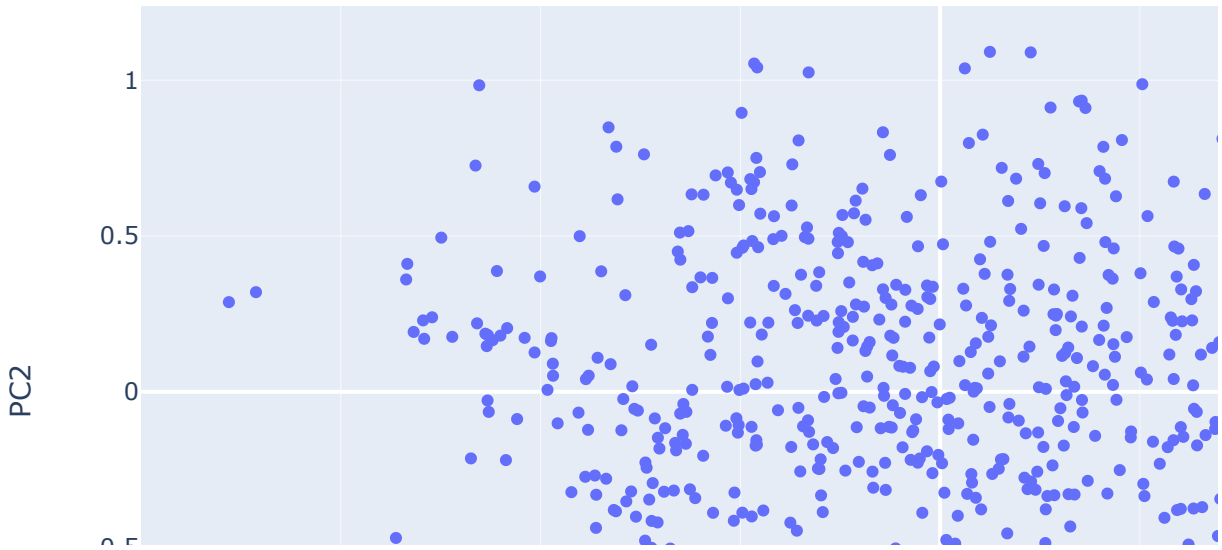
Out[9]:

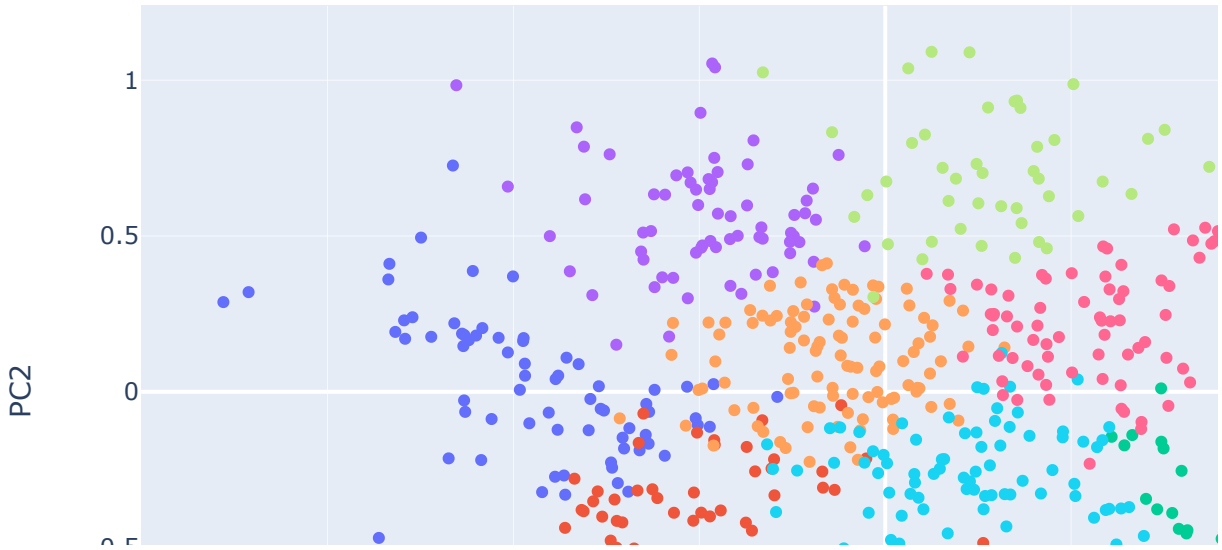
WIN%	0.696570
PTS	114.994560
FGM	39.739010
FGA	88.585854
FG%	47.687645
3PM	11.218748
3PA	26.364664
3P%	37.536713
FTM	22.172733
FTA	30.484396
FT%	78.387138
OREB	11.728777
DREB	35.108219
REB	46.610382
AST	25.787374
TOV	15.358849
STL	7.638455
BLK	6.857519
BLKA	4.534201
PF	21.939579
PFD	21.415867
+/-	6.341363
OFFRTG	112.988861
DEFRTG	108.450165
NETRTG	7.899961
AST%	63.376869
AST/TO	1.644425
AST\nRATIO	18.018234
OREB%	31.595293

```
DREB%           73.153694
REB%            52.549057
TOV%            14.133371
EFG%            53.588612
TS%             57.477962
PACE           102.560585
PIE             57.111938
POSS          8750.960938
dtype: float32
```

In [10]:
```
nba_analysis.k_nearest_neighbors(team,minutes_selection_method='average',st
```

```
Cluster: 3
```





Out[10]:

	TEAM	YEAR	WIN%
587	Philadelphia 76ers	2018-19	0.622
593	Utah Jazz	2018-19	0.610
247	Los Angeles Lakers	2007-08	0.695
607	Los Angeles Clippers	2019-20	0.681
241	Denver Nuggets	2007-08	0.610