Name: Yousef Al-ghanim

ID: 2122123716

Midterm

Results:

1-Variable identification:

Team Dataset:

Variable	Variable definition	Data type	Missing data report	Report on the distribution of the data	level of analysis
year	The year in which the team stats was collected	Continuous(Int64)	0 missing data	The maximum value for the year column is 2011 and the minimum is 1937.	Team
lgID	league ID	Categorical(string)	0 missing data	This column has 6 unique values and the NBA is the most frequent value 1304 (84.9%)	League
tmID	Team ID	Categorical(string)	0 missing data	This column has 161 unique values	Team
franchID	franchise ID	Categorical(string)	0 missing data	This column has 104 unique values and DET is the most freuqnt value 71	Franchise
confID	Confrence ID (east or west)	Categorical(string)	472 missing data	This column has only 2 values WC (West confrence) and EC (East confrence)	Confrence
divID	division ID	Categorical(string)	38 missing value	This column has 13 unique values	Division
rank	the rank of the team in the division	Discrete(Int64)	1 missing data	This column has 10 unique values with value 1 and 2 is the most frequent values (18.55%)	Division
confRank	The rank of the team in the confrence	Discrete(Int64)	562 missing data	This column has 15 unique values	Confrence
playoff	Where did the team reach in the play (final, semi-final,etc..)	Categorical(string)	635 missing data	This column has 15 unique values With C1 (Lost conference 1st round) being the most frequent value	Playoff

Variable	Variable definition	Data type	Missing data report	Report on the distribution of the data	level of analysis
name	The name of the team	Categorical(string)	0 missing data	This column has 156 unique values	Team
o_fgm	Offense field goal made	Continuous(Int64)	10 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
o_fga	Offense field goal attempted	Continuous(Int64)	125 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
o_ftm	Offense free throw made	Continuous(Int64)	10 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
o_fta	Offense free throw attempted	Continuous(Int64)	76 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
o_3pm	Offense 3 point made	Continuous(Int64)	534 missing data	The maximum value for the o_3pm column is 841 and the minimum is 0.	Team
o_3pa	Offense 3 point attempted	Continuous(Int64)	534 missing data	The maximum value for the o_3pa column is 2283 and the minimum is 0.	Team
o_oreb	Offense Offensive rebound	Continuous(Int64)	473 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
o_dreb	Offense Defensive rebound	Continuous(Int64)	473 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
o_reb	Offense rebound	Continuous(Int64)	173 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
o_asts	Offense assists	Continuous(Int64)	125 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
o_pf	Offense personal fouls	Continuous(Int64)	107 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
o_stl	Offense steals	Continuous(Int64)	494 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
o_to	Offense turnovers	Continuous(Int64)	429 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team

Variable	Variable definition	Data type	Missing data report	Report on the distribution of the data	level of analysis
o_blk	Offense blocks	Continuous(Int64)	494 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
o_pts	Offense points	Continuous(Int64)	1 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
d_fgm	Defense field goal made	Continuous(Int64)	362 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
d_fga	Defense field goal attempted	Continuous(Int64)	378 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
d_ftm	Defense free throw made	Continuous(Int64)	362 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
d_fta	Defense free throw attempted	Continuous(Int64)	532 missing data	The maximum value for the d_fta column is 2285 and the minimum is 12.	Team
d_3pm	Defense 3 point made	Continuous(Int64)	548 missing data	After ploting the values of this column using histogram we can say that it's Bimodel distribution (two peaks)	Team
d_3pa	Defense 3 point attempted	Continuous(Int64)	378 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
d_oreb	Defense Offensive rebound	Continuous(Int64)	473 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
d_dreb	Defense Defensive rebound	Continuous(Int64)	473 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
d_reb	Defense rebound	Continuous(Int64)	378 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
d_asts	Defense assists	Continuous(Int64)	378 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
d_pf	Defense personal fouls	Continuous(Int64)	362 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team

Variable	Variable definition	Data type	Missing data report	Report on the distribution of the data	level of analysis
d_stl	Defense steals	Continuous(Int64)	494 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
d_to	Defense turnovers	Continuous(Int64)	429 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
d_blk	Defense blocks	Continuous(Int64)	494 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
d_pts	Defense points	Continuous(Int64)	1 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
o_tmRebound	Offense team rebound	Continuous(Int64)	1360 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
d_tmRebound	Defense team rebound	Continuous(Int64)	1521 missing data	The maximum value for the d_tmRebound column is 769 and the minimum is 385.	Team
homeWon	Games won at home arena	Continuous(Int64)	107 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
homeLost	Games lost at home arena	Continuous(Int64)	110 missing data	After ploting the values of this column using histogram we can say that it's skewed to the right	Team
awayWon	Games won at the arena of the opponent team	Continuous(Int64)	129 missing data	After ploting the values of this column using histogram we can say that it's skewed to the right	Team
awayLost	Games lost at the arena of the opponent team	Continuous(Int64)	107 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
neutWon	Games won at neural arena	Continuous(Int64)	0 missing data	The maximum value for the neutWon column is 18 and the minimum is 0.	Team
neutLoss	Games lost at neural arena	Continuous(Int64)	0 missing data	The maximum value for the neutLoss column is 23 and the minimum is 0.	Team
confWon	Games won at conference	Continuous(Int64)	0 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
confLoss	Games lost at conference	Continuous(Int64)	0 missing data	After ploting the values of this column using histogram we can say that it's skewed to the right	Team

Variable	Variable definition	Data type	Missing data report	Report on the distribution of the data	level of analysis
divWon	Games won at division	Continuous(Int64)	237 missing data	After ploting the values of this column using histogram we can say that it's skewed to the right	Team
divLoss	Games lost at division	Continuous(Int64)	237 missing data	After ploting the values of this column using histogram we can say that it's skewed to the right	Team
pace	Possessions per game	Continuous(Int64)	1446 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
won	Games won in the season	Continuous(Int64)	2 missing data	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
lost	Games lost in the season	Continuous(Int64)	2 missing data	After ploting the values of this column using histogram we can say that it's normaly distributed (bell shape)	Team
games	Total games played	Continuous(Int64)	0 missing data	The maximum value for the games column is 84 and the minimum is 2.	Team
min	Minitues played in the season	Continuous(Float)	214 missing data	The maximum value for the min column is 20460.0 and the minimum is 2640.0.	Team
arena	The name of the arena	Categorical(string)	189 missing data	This column has 165 unique values and the Boston Garden is the most frequent value 49.	Team

Clarification: Offense stats stands for team stats and Defense stats stands for opponents stats, for example Chicago Bulls has offense field goal made of 2467 and defense field goal made of 2284, this means that Chicago Bulls made 2467 field goal and their oppentents collectively made 2284 field goal.

Click here for the abbreviations

Draft Dataset:

Variable	Variable definition	Data type	Missing data report	Report on the distribution of the data	level of analysis
draftYear	The year of the draft	Continuous(Int64)	0 missing data	The maximum value for the draftYear column is 2011 and the minimum is 1947.	Draft
draftRound	Which round did the player got drafted	Discrete(Int64)	1493 missing data	This column has 21 unique values and value 1 and 2 are the most frequent values	Draft

Variable	Variable definition	Data type	Missing data report	Report on the distribution of the data	level of analysis
draftSelection	Player selection in his round (first to be selected, second, third, etc..)	Discrete(Int64)	2184 missing data	This column has 31 unique value with maximum value of 31 and minimum value of 1	Draft
draftOverall	Overall selection for the players	Discrete(Int64)	2167 missing data	The maximum value for the draftOverall column is 239 and the minimum is 1.	Draft
tmID	Team ID	Categorical(string)	0 missing data	This column has 96 unique values	Team
firstName	Player first name	Categorical(string)	0 missing data	This column has 1465 unique values with John being the most frequent first name	Player
lastName	Player Last name	Categorical(string)	0 missing data	This column has 4193 unique values with Smith being the most frequent last name	Player
playerID	Player ID	Categorical(string)	5189 missing data	This column has 3133 unique values	Player
draftFrom	From where did the player get drafted	Categorical(string)	6 missing data	This column has 1266 unique values and UCLA being the most frequent value	Draft
lgID	League ID	Categorical(string)	0 missing data	This column has 2 unique values with NBA being the most frequent value 7559 (83.96%)	League

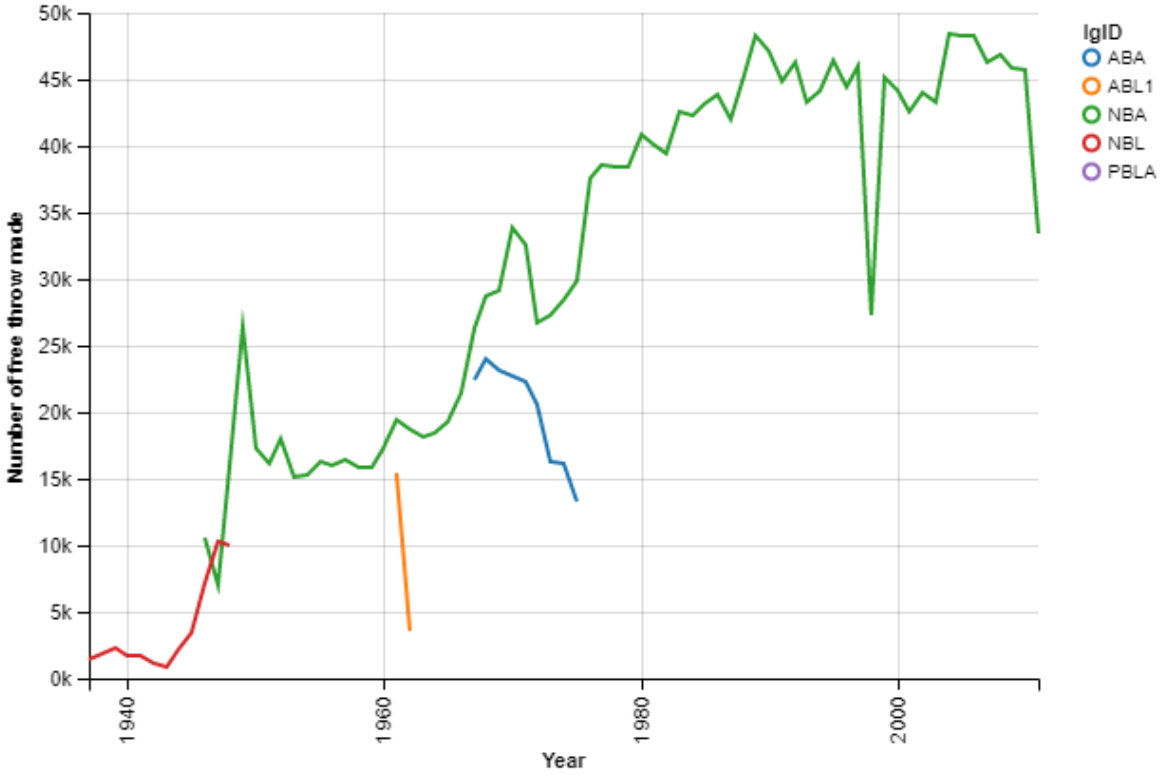
2-Transformed variables in the data set:

Variable	Variable description	Steps in transformation	Distribution	level of analysis
draftCountry	The country that the player has been drafted from	By using regular expression and fetching the values between the parentheses then checking if the value is a country or not	This column has 30 unique values with the USA being the most frequent value	Draft
o_ftm_games_ratio	The ratio between the free throw made and the played games	By dividing the o_ftm by the games column and assinging the values to o_ftm_games_ratio	After ploting the values of this column using histogram we can say that it's skewed to the left	Team
o_fgp	The team percentage of field goal made and attempted	By dividing the o_fgm column by o_fga column	After ploting the values of this column using histogram we can say that it's skewed to the left	Team

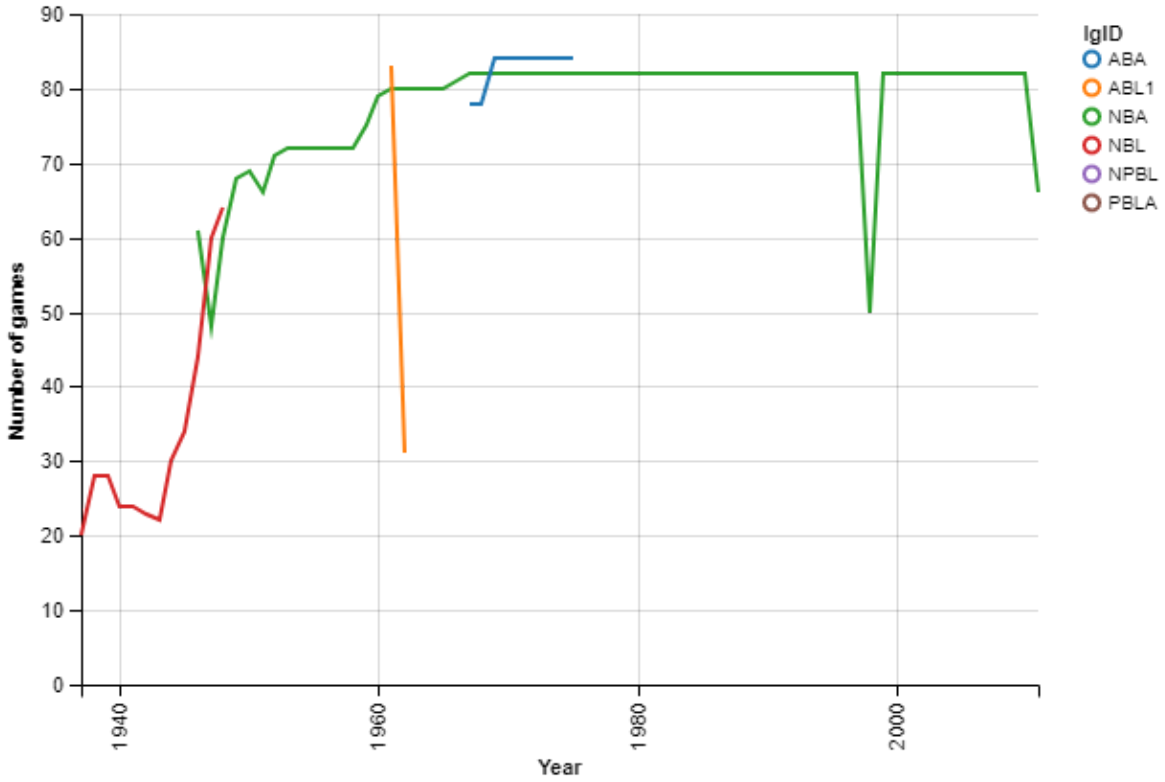
Variable	Variable description	Steps in transformation	Distribution	level of analysis
teamDefensiveRating	Defensive rating of the team (Less means better)	By dividing d_pts by pace	Not applicable	Team
wonLostPercentage	the percentage of won games out of overall played games	By dividing the won column over games column	After ploting the values of this column using histogram we can say that it's skewed to the left	Team

3- Insights:

1. from the following graph we can say that basketball players are getting better in shooting the free throws as time goes on,



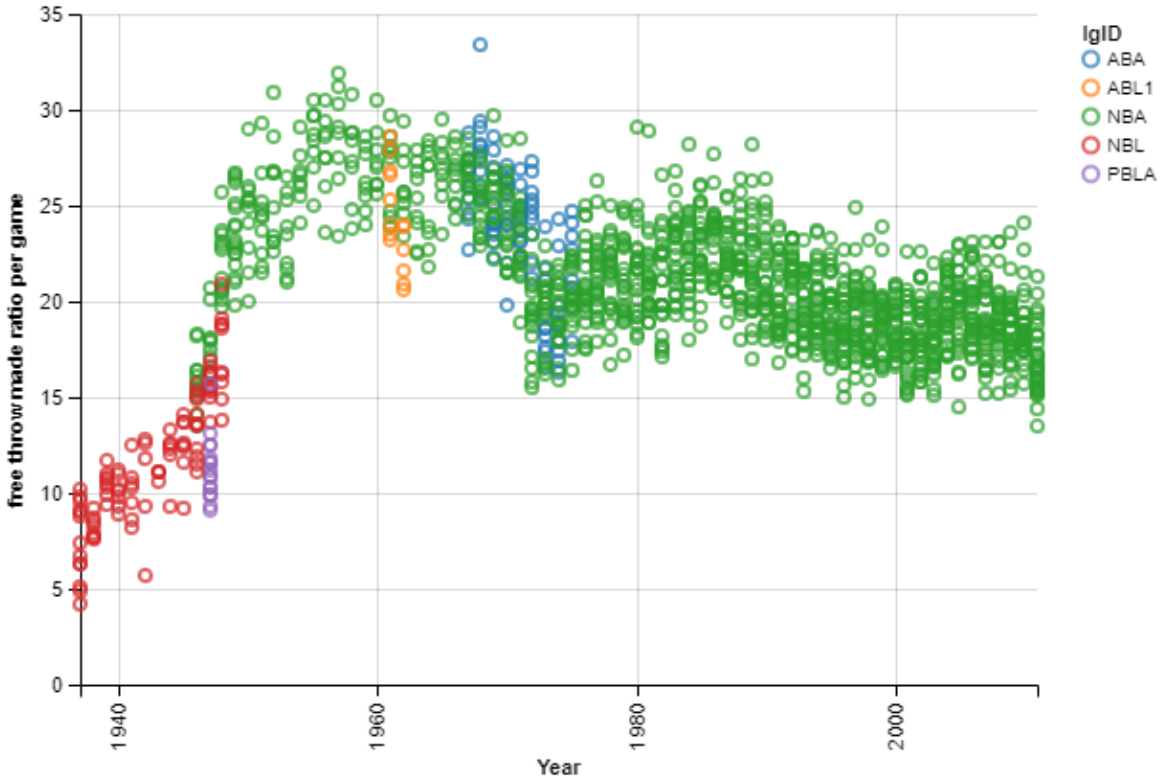
but from the graph you can see that there is a dip in the years between 1990 and 2000 why is that?
Well the reason for that is in the 1998 the league decided to change the number of games in the season from 82 to 50 which means that the players has less opportunities to shoot free throws.



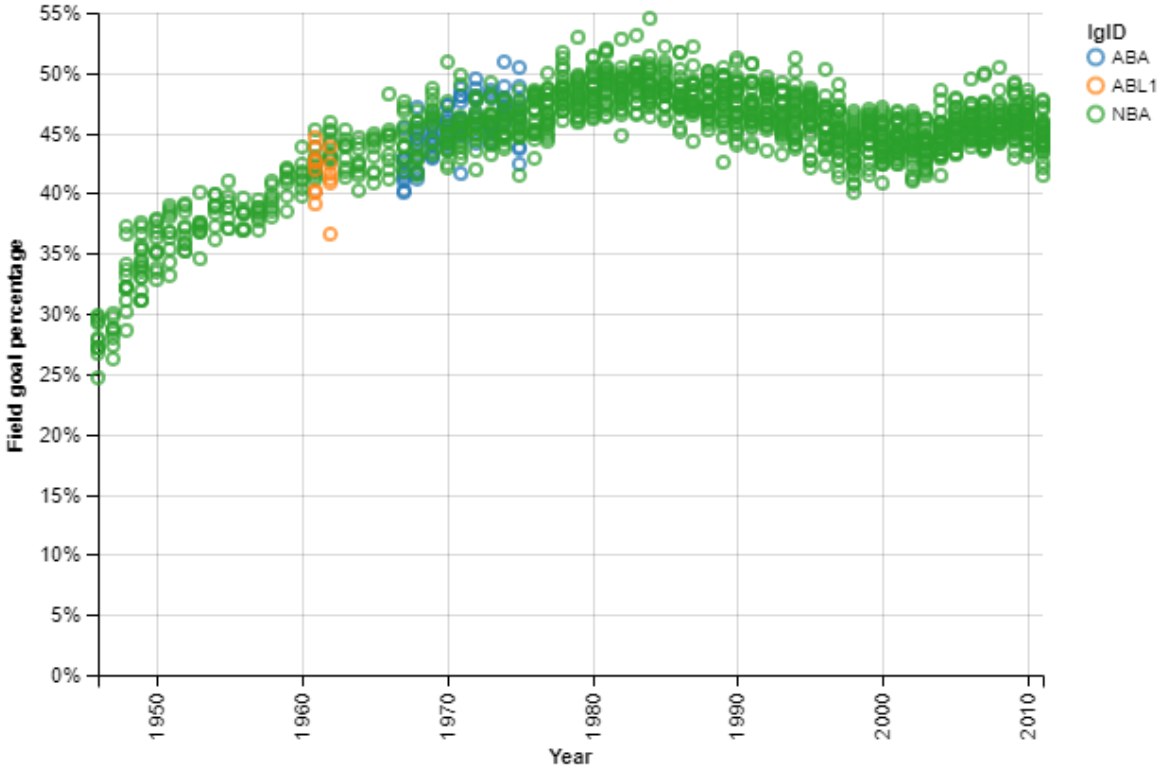
So we need to find the ratio of free throw made per game,

and from the graph we can see that the sum of free throw shooting ratio per game is getting better and better, but are they?

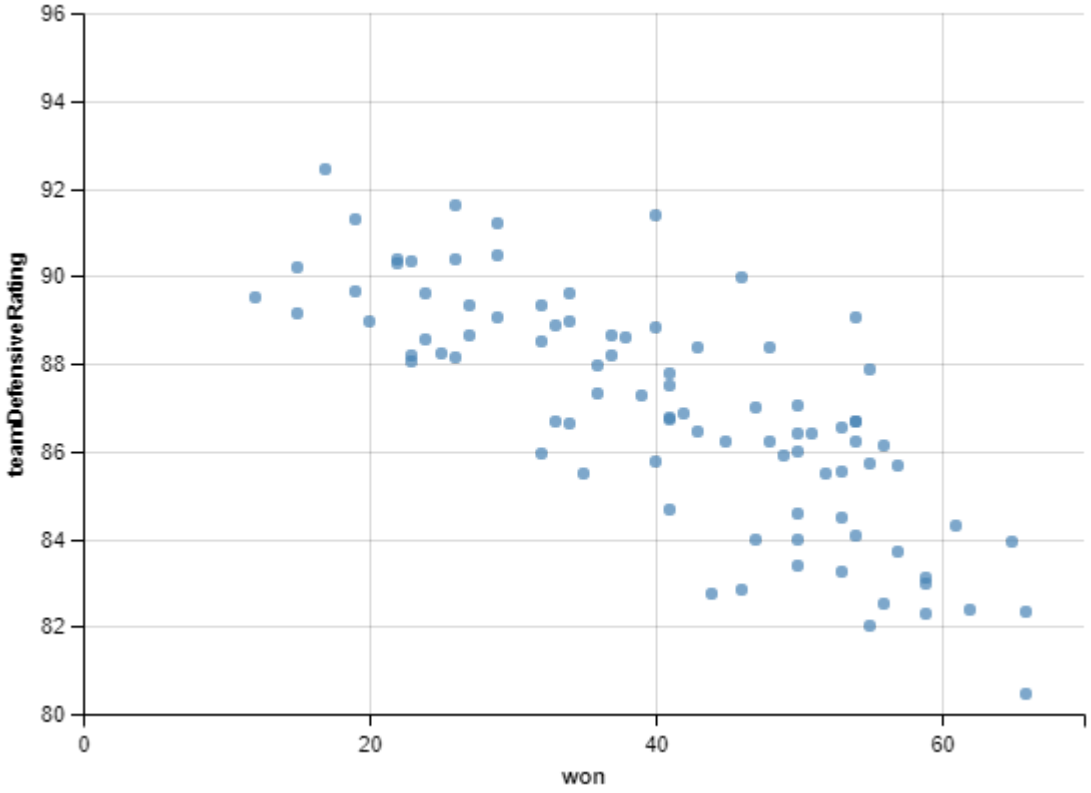
No they are not, because we didn't take the number of teams in to account, and here is the true representation of the free throw ratio per game.



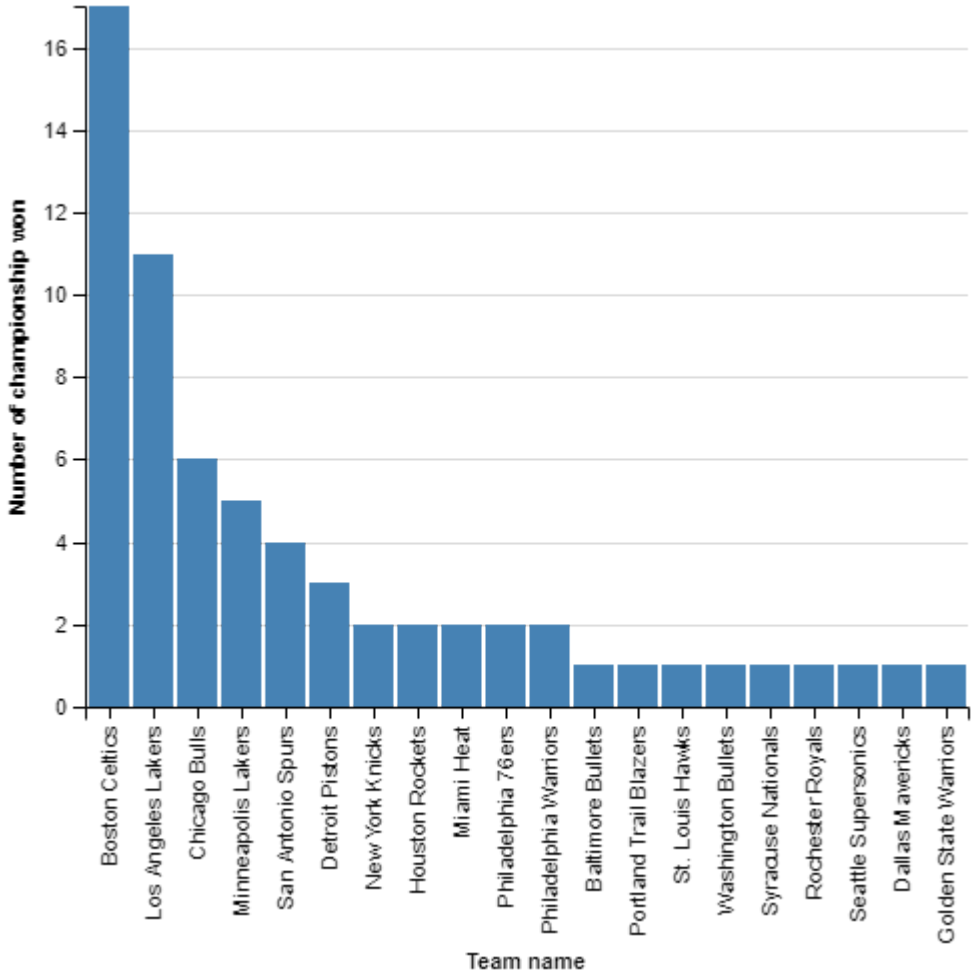
2. The basketball players has become more efficient in shoot field goals:



3. There is a negative relationship and strong negative correlation (-0.787) between team defensive rating and how many time did the team win, and since the less the value of defensive rating the better, therefore you will have a better change of winning if you have better defense.



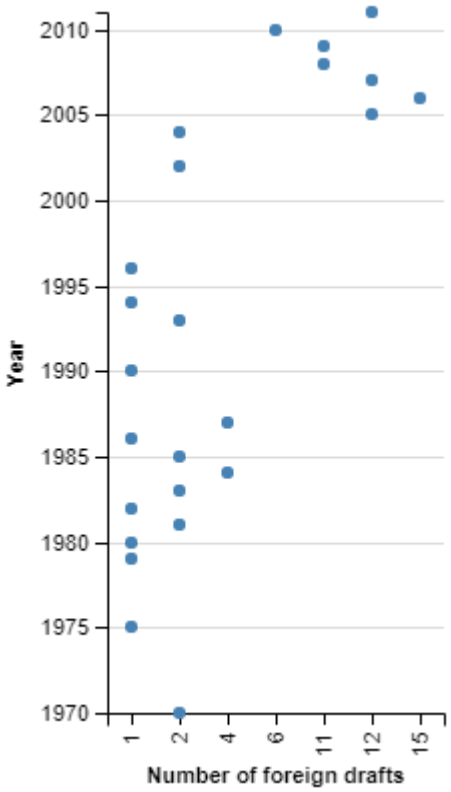
4. Boston Celtics has won the most NBA championship (17) followed by Los Angeles Lakers



5. 1995 Chicago Bulls has the highest win to lose ratio (87.8%) and they won the championship that year.

4- Questions:

1. Why between the year 2005 and 2010 the number of foreign drafts has increased?(maybe the league has become more attractive to foreign players, interms of the income they will recive?)



2. On what basis are the teams choosing their drafts? (maybe if have drafted players stats we can answer this question)
3. Why the the sum of home won games (32811) is greater than the sum of away won games (19368)? (does the home crowd effect players performance?)

Analysis:

In [874]:
```
import pandas as pd
import altair as alt
import numpy as np
import re
import requests
alt.Chart.max_rows = 10000
%matplotlib inline
#players_df=pd.read_csv('basketball_players.csv',low_memory=False)
teams_df = pd.read_csv('basketball_teams.csv',low_memory=False)
drafts_df = pd.read_csv('basketball_draft.csv',low_memory=False)
abbrev_df = pd.read_csv('basketball_abbrev.csv', low_memory=False)
# To show all column in the dataframe when I print it or  call for head/tail metho
pd.set_option('display.max_columns', None)
```

In [875]:
```
teams_df.head()
```

Out[875]:

	year	lgID	tmID	franchID	confID	divID	rank	confRank	playoff	name	o_fgm	o_fga	o_
0	1946	NBA	BOS	BOS	NaN	ED	5	0	NaN	Boston Celtics	1397	5133	
1	1946	NBA	CHS	CHS	NaN	WD	1	0	F	Chicago Stags	1879	6309	
2	1946	NBA	CLR	CLR	NaN	WD	3	0	R1	Cleveland Rebels	1674	5699	
3	1946	NBA	DTF	DTF	NaN	WD	4	0	NaN	Detroit Falcons	1437	5843	
4	1946	NBA	NYK	NYK	NaN	ED	3	0	SF	New York Knicks	1465	5255	

In [876]:
```
drafts_df.head()
```

Out[876]:

	draftYear	draftRound	draftSelection	draftOverall	tmID	firstName	lastName	suffixName	play
0	1967	0	0	0	ANA	Darrell	Hardy	NaN	hardy
1	1967	0	0	0	ANA	Bob	Krulish	NaN	
2	1967	0	0	0	ANA	Bob	Lewis	NaN	lewis
3	1967	0	0	0	ANA	Mike	Lynn	NaN	lynn
4	1967	0	0	0	ANA	Tom	Workman	NaN	workn

In [877]:
```
# All the abbreviations in the teams dataset
abbrev_df
```

Out[877]:

	abbrev_type	code	full_name
0	Round	DF	Division Finals
1	Conference	EC	Eastern Conference
2	Conference	WC	Western Conference
3	Division	AT	Atlantic Division
4	Division	ED	Eastern Division
5	Division	CD	Central Division
6	Division	WD	Western Division
7	Division	SE	Southeast Division
8	Division	MW	Midwest Division
9	Division	NW	Northwest Division
10	Division	SW	Southwest Division
11	Division	PC	Pacific Division
12	Playoffs	R1	Lost round 1
13	Playoffs	SF	Lost semi-finals
14	Playoffs	F	Lost finals
15	Playoffs	NC	Won NBA championship
16	Playoffs	AC	Won ABA championship
17	Playoffs	DT	Lost division tie
18	Playoffs	DF	Lost division finals
19	Playoffs	DS	Lost division semi-finals
20	Playoffs	DR	Lost division round-robin
21	Playoffs	CF	Lost conference finals
22	Playoffs	CS	Lost conference semi-finals
23	Playoffs	C1	Lost conference 1st round
24	Round	DT	Division Tiebreaker
25	Round	DSF	Division Semifinals
26	Round	CF	Conference Finals
27	Round	CFR	Conference First Round
28	Round	CSF	Conference Semifinals
29	Round	RR	Round Robin
30	Round	F	Finals
31	Round	QF	Quarterfinals
32	Round	SF	Semifinals

	abbrev_type	code	full_name
33	Round	FR	First Round
34	Division	EA	East Division
35	Division	WE	West Division
36	Playoffs	WC	Won Championship
37	Playoffs	LC	Lost Championship
38	Division	SO	South Division
39	Division	NO	North Division

In [878]:
```
# removing attendance column due ambiguous data, there is only two different valu
del teams_df['attendance']

# removeing bbtmID column due to redundancy.
del teams_df['bbtmID']

# removing suffixName column from drafts_df because it has only 2 none null value
del drafts_df['suffixName']
```

In [879]:
```
# checking if the columns has been deleted
teams_df.head()
```

Out[879]:

	year	lgID	tmID	franchID	confID	divID	rank	confRank	playoff	name	o_fgm	o_fga	o_
0	1946	NBA	BOS	BOS	NaN	ED	5	0	NaN	Boston Celtics	1397	5133	
1	1946	NBA	CHS	CHS	NaN	WD	1	0	F	Chicago Stags	1879	6309	
2	1946	NBA	CLR	CLR	NaN	WD	3	0	R1	Cleveland Rebels	1674	5699	
3	1946	NBA	DTF	DTF	NaN	WD	4	0	NaN	Detroit Falcons	1437	5843	
4	1946	NBA	NYK	NYK	NaN	ED	3	0	SF	New York Knicks	1465	5255	

In [880]:
```
# checking if the column has been deleted
drafts_df.head()
```

Out[880]:

	draftYear	draftRound	draftSelection	draftOverall	tmID	firstName	lastName	playerID	draftFr
0	1967	0	0	0	ANA	Darrell	Hardy	hardyda01	Ba
1	1967	0	0	0	ANA	Bob	Krulish	NaN	Pa
2	1967	0	0	0	ANA	Bob	Lewis	lewisbo01	N Carc
3	1967	0	0	0	ANA	Mike	Lynn	lynnmi01	U(
4	1967	0	0	0	ANA	Tom	Workman	workmto01	Sea

In [881]:
```
# checking for the missing data in year column in teams_df:
teams_df.year.isnull().sum()
```

Out[881]: 0

In [882]:
```
# Checking for the distribution of year column in teams_df:
print('The maximum value for the year column is {} and the minimum is {}.'.format
                        teams_df.year.max(), teams_df.year.min()))
```

```
The maximum value for the year column is 2011 and the minimum is 1937.
```

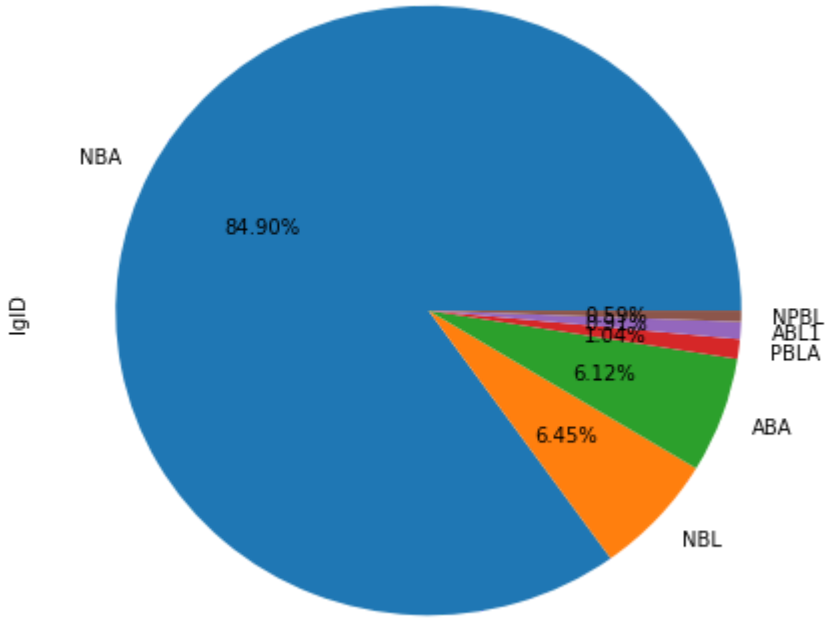
In [883]:
```
# checking for the missing data in lgID column in teams_df:
teams_df.lgID.isnull().sum()
```

Out[883]: 0

In [884]:
```
# Checking for the distribution of lgID column in teams_df:
print(teams_df.lgID.value_counts())
teams_df.lgID.value_counts().plot(kind='pie', autopct='%1.2f%%',figsize=[7,7])
```

```
NBA     1304
NBL       99
ABA       94
PBLA      16
ABL1      14
NPBL       9
Name: lgID, dtype: int64
```

Out[884]: <matplotlib.axes._subplots.AxesSubplot at 0x14b31bb6d30>



In [885]:
```
# checking for the missing data in tmID column in teams_df:
teams_df.tmID.isnull().sum()
```

Out[885]: 0

In [886]:
```
# Checking for the distribution of tmID column in teams_df:
print(teams_df.tmID.value_counts().head(),
      '\n',
     'Total number of uniuqe values is {}'.format(teams_df.tmID.value_counts().co
```

```
BOS    66
NYK    66
DET    55
LAL    52
PHI    49
Name: tmID, dtype: int64
 Total number of uniuqe values is 161
```

In [887]:
```
# checking for the missing data in franchID column in teams_df:
teams_df.franchID.isnull().sum()
```

Out[887]: 0

In [888]:
```
# Checking for the distribution of franchID column in teams_df:
print(teams_df.franchID.value_counts().head(),
      '\n',
     'Total number of uniuqe values is {}'.format(teams_df.franchID.value_counts(
```

```
DET    71
SAC    67
PHI    66
NYK    66
ATL    66
Name: franchID, dtype: int64
 Total number of uniuqe values is 104
```

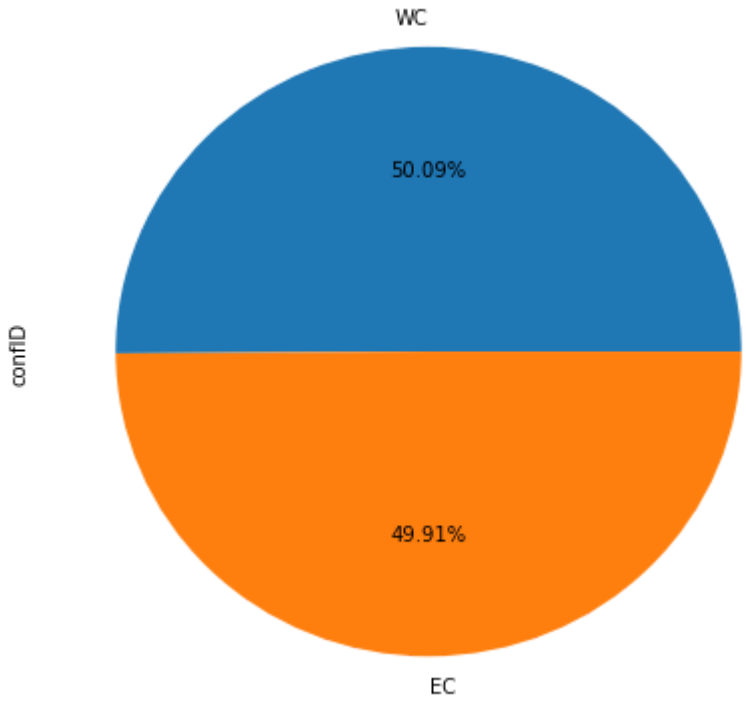
In [889]:
```
# checking for the missing data in confID column in teams_df:
teams_df.confID.isnull().sum()
```

Out[889]: 472

In [890]:
```
# Checking for the distribution of confID column in teams_df:
print(teams_df.confID.unique())
teams_df.confID.value_counts().plot(kind='pie', autopct='%1.2f%%',figsize=[7,7])
```

```
[nan 'EC' 'WC']
```

Out[890]: <matplotlib.axes._subplots.AxesSubplot at 0x14b31ae4278>



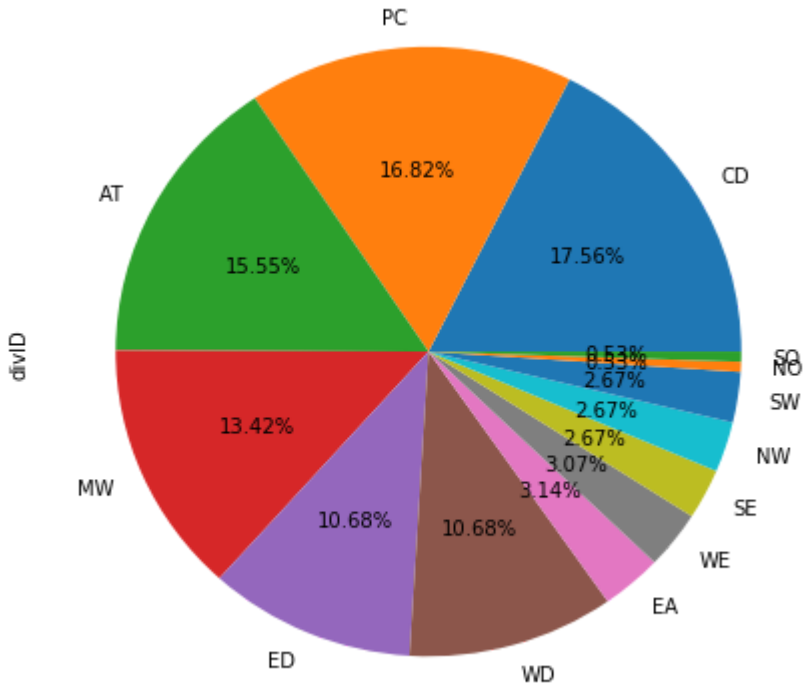
In [891]:
```
# checking for the missing data in divID column in teams_df:
teams_df.divID.isnull().sum()
```

Out[891]: 38

In [892]:
```
# Checking for the distribution of divID column in teams_df:
print(teams_df.divID.unique())
teams_df.divID.value_counts().plot(kind='pie', autopct='%1.2f%%',figsize=[7,7])
```

```
['ED' 'WD' 'CD' 'AT' 'MW' 'PC' nan 'SE' 'SW' 'NW' 'EA' 'WE' 'SO' 'NO']
```

Out[892]: <matplotlib.axes._subplots.AxesSubplot at 0x14b31ab7908>



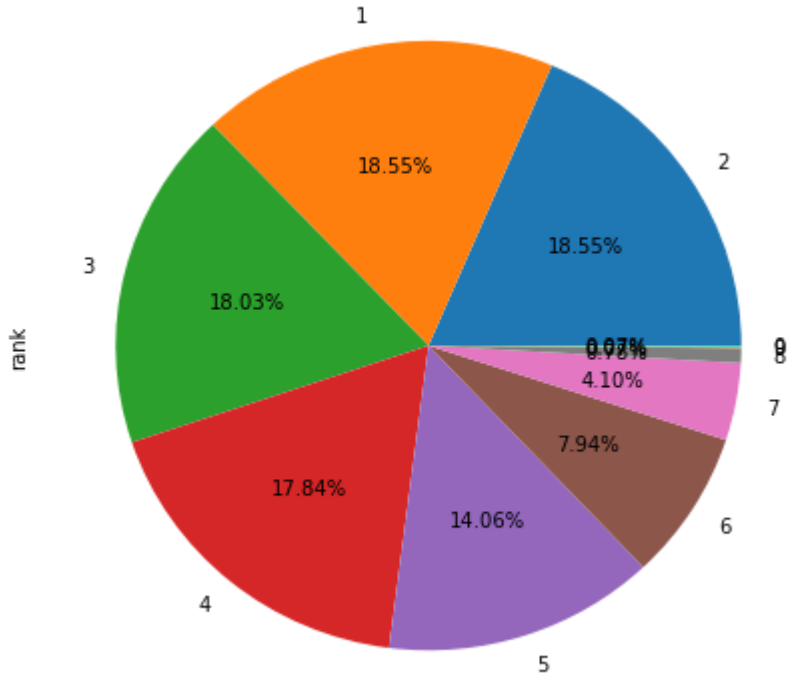
In [893]:
```
# checking for the missing data in rank column in teams_df:
len(teams_df[(teams_df['rank'].isnull() == True) | (teams_df['rank'] == 0)])
```

Out[893]: 1

In [894]:
```
# Checking for the distribution of rank column in teams_df:
print(teams_df['rank'].unique())
teams_df['rank'].value_counts().plot(kind='pie', autopct='%1.2f%%',figsize=[7,7])
# from this graph we can tell the some leagues don't have ranks numbers more than
# and/or there has been change in the divisions to be no more than 5 teams in one
```

```
[5 1 3 4 2 6 7 8 9 0]
```

Out[894]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x14b319aea58>
```



In [895]:
```
# checking if the theory above is true:
teams_df[(teams_df['rank'] > 5) & (teams_df.year <= 2003) & (teams_df.lgID == 'NB
```

Out[895]:

	year	lgID	tmID	franchID	confID	divID	rank	confRank	playoff	name	o_fgm	o_fg
1148	2003	NBA	ORL	ORL	EC	AT	7	15	NaN	Orlando Magic	2904	676
1150	2003	NBA	PHO	PHO	WC	PC	6	13	NaN	Phoenix Suns	2958	667
1155	2003	NBA	TOR	TOR	EC	CD	6	10	NaN	Toronto Raptors	2654	634
1156	2003	NBA	UTA	UTA	WC	MW	7	9	NaN	Utah Jazz	2690	617
1157	2003	NBA	WAS	WAS	EC	AT	6	13	NaN	Washington Wizards	2758	655

In [896]:
```
teams_df[(teams_df['rank'] > 5) & (teams_df.year > 2003) & (teams_df.lgID == 'NBA
```

Out[896]:

	year	lgID	tmID	franchID	confID	divID	rank	confRank	playoff	name	o_fgm	o_fga	o_ftm

In [897]:
```
teams_df[(teams_df['rank'] > 8)  & (teams_df.lgID == 'NBA')].tail()
```

Out[897]:

	year	lgID	tmID	franchID	confID	divID	rank	confRank	playoff	name	o_fgm	o_fga	o_ftm

In [898]:
```
# It seems that our theory is correct.
```

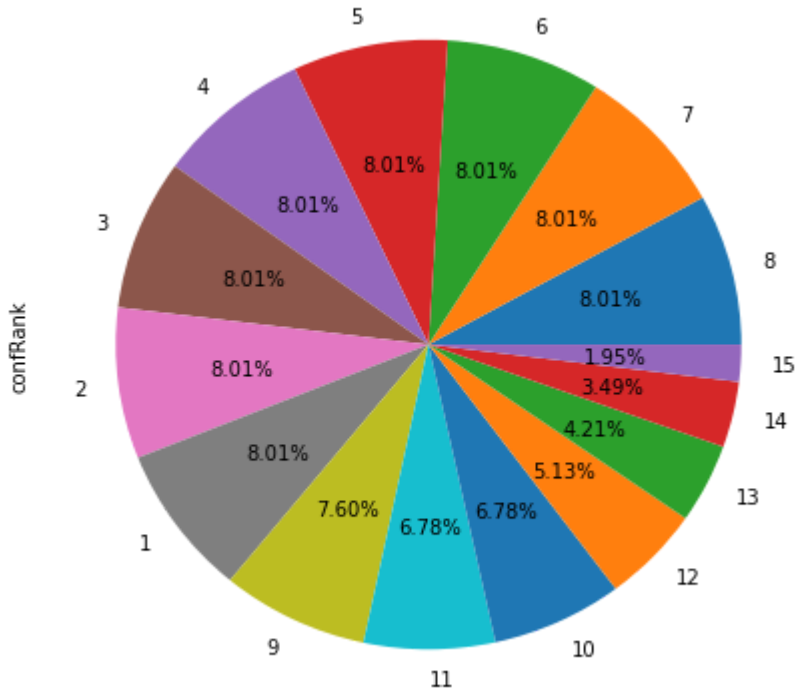
In [899]:
```
# checking for the missing data in confRank column in teams_df:
len(teams_df[(teams_df.confRank.isnull() == True) | (teams_df.confRank == 0)])
```

Out[899]: 562

In [900]:
```
# Checking for the distribution of rank column in teams_df:
print(teams_df.confRank[(teams_df.confRank.isnull() == True) | (teams_df.confRank
teams_df.confRank[(teams_df.confRank.isnull() == True) | (teams_df.confRank != 0)
```

```
[ 4  3  5  7  2  6  8  1  9 10 11 12 13 14 15]
```

Out[900]: <matplotlib.axes._subplots.AxesSubplot at 0x14b318f8668>



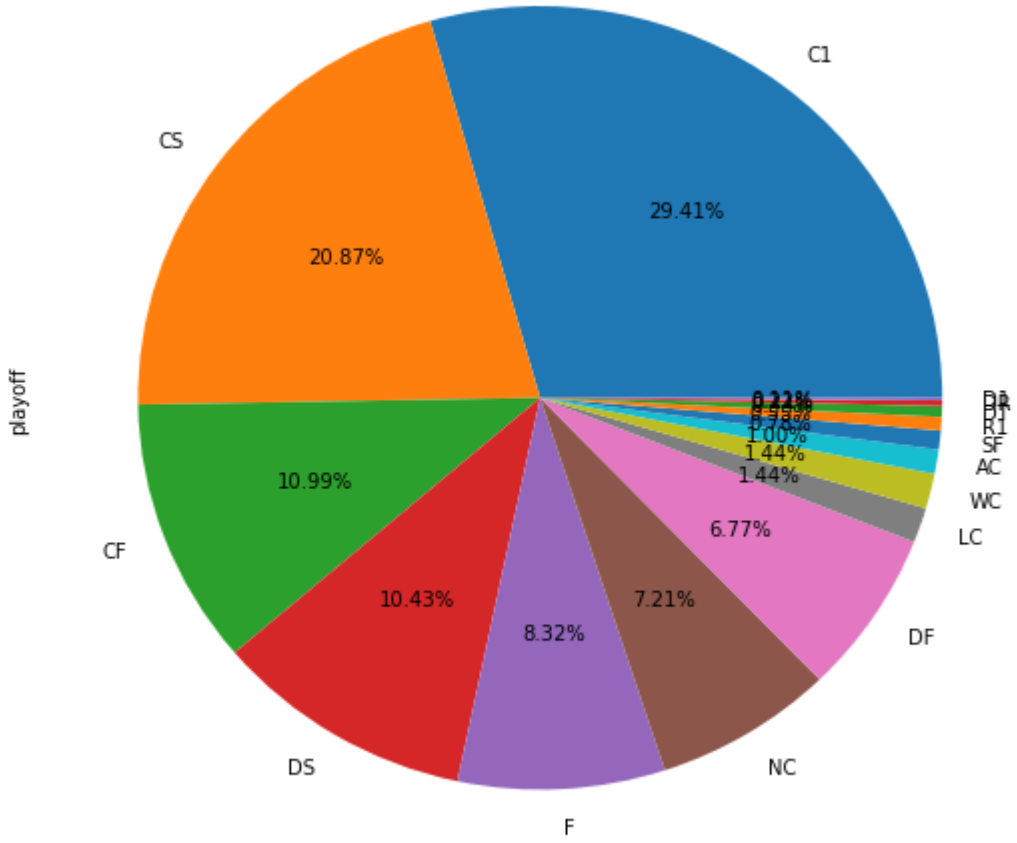
In [901]:
```
# checking for the missing data in playoff column in teams_df:
teams_df.playoff.isnull().sum()
```

Out[901]: 635

In [902]:
```
# Checking for the distribution of playoff column in teams_df:
print(teams_df.playoff.unique())
teams_df.playoff.value_counts().plot(kind='pie', autopct='%1.2f%%',figsize=[9,9])
```

```
[nan 'F' 'R1' 'SF' 'NC' 'DT' 'DS' 'DF' 'DR' 'AC' 'CS' 'CF' 'C1' 'D1' 'WC'
 'LC']
```

Out[902]: <matplotlib.axes._subplots.AxesSubplot at 0x14b313ad048>



In [903]:
```
# checking for the missing data in name column in teams_df:
teams_df.name.isnull().sum()
```

Out[903]: 0

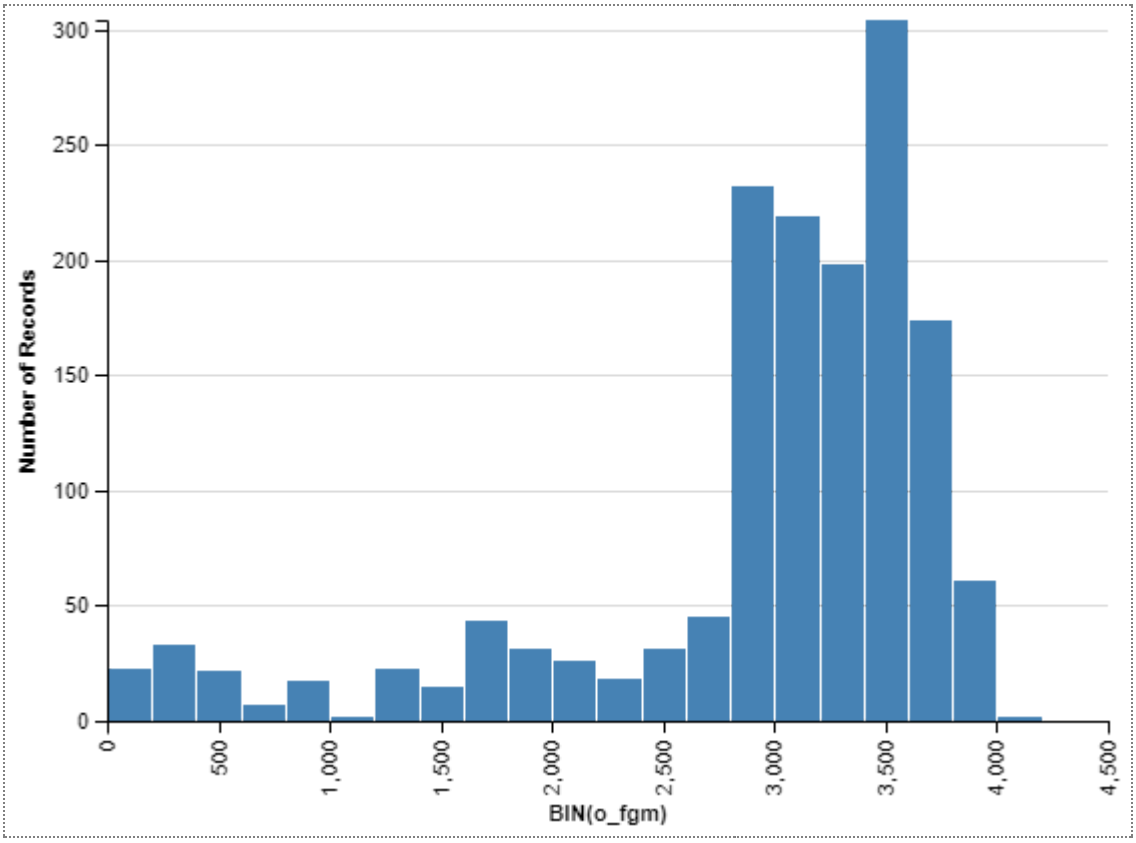
In [904]:
```
# Checking for the distribution of name column in teams_df:
teams_df.name.value_counts().head()
```

Out[904]:
```
Boston Celtics        66
New York Knicks       66
Detroit Pistons       55
Los Angeles Lakers    52
Philadelphia 76ers    49
Name: name, dtype: int64
```

In [905]:
```
# checking for the missing data in o_fgm column in teams_df:
len(teams_df[(teams_df.o_fgm.isnull() == True) | (teams_df.o_fgm == 0)])
```

Out[905]: 10

In [906]:
```
# Checking for the distribution of o_fgm column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_fgm.notnull()) & (teams_df.o_fgm !=
alt.Chart(teams_df[(teams_df.o_fgm.notnull()) & (teams_df.o_fgm != 0)]).mark_bar(
    x=alt.X('o_fgm', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

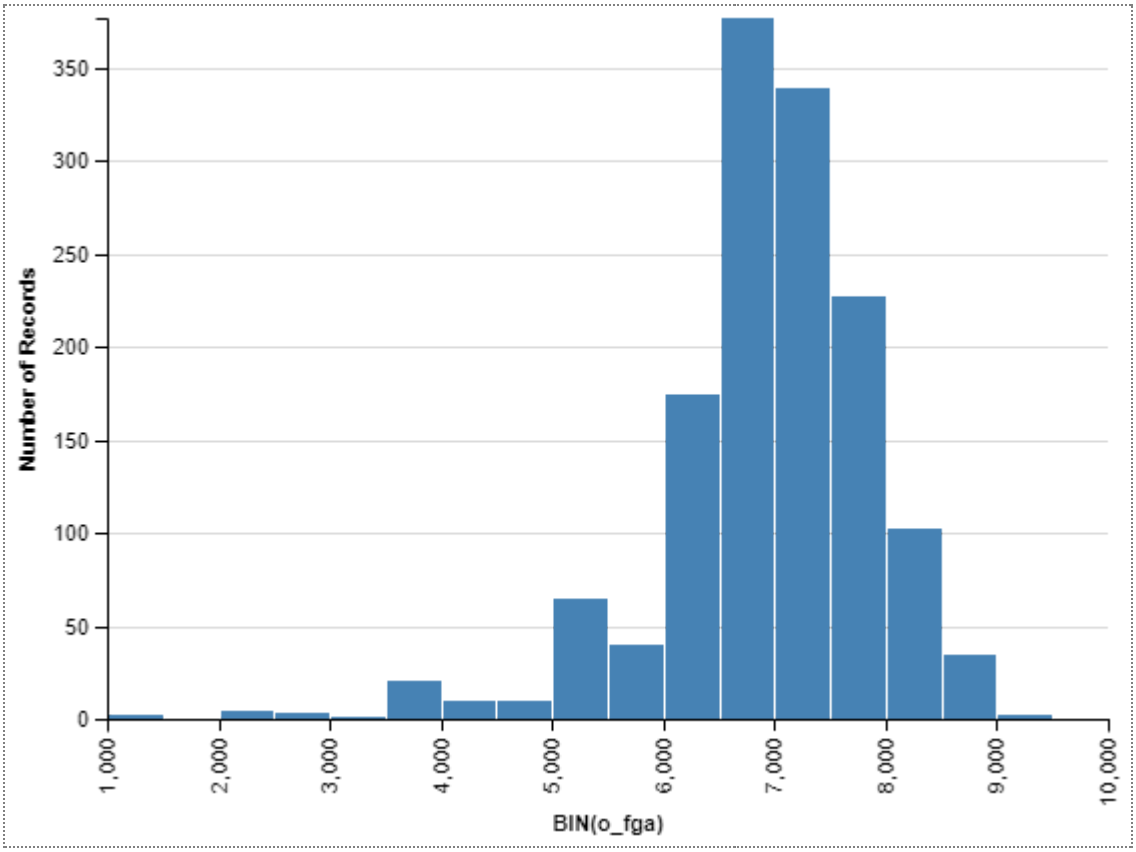


Export as PNG View Source Open in Vega Editor

In [907]:
```
# checking for the missing data in o_fga column in teams_df:
len(teams_df[(teams_df.o_fga.isnull() == True) | (teams_df.o_fga == 0)])
```

Out[907]: 125

In [908]:
```
# Checking for the distribution of o_fga column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_fga.notnull()) & (teams_df.o_fga !=
alt.Chart(teams_df[(teams_df.o_fga.notnull()) & (teams_df.o_fga != 0)]).mark_bar(
    x=alt.X('o_fga', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

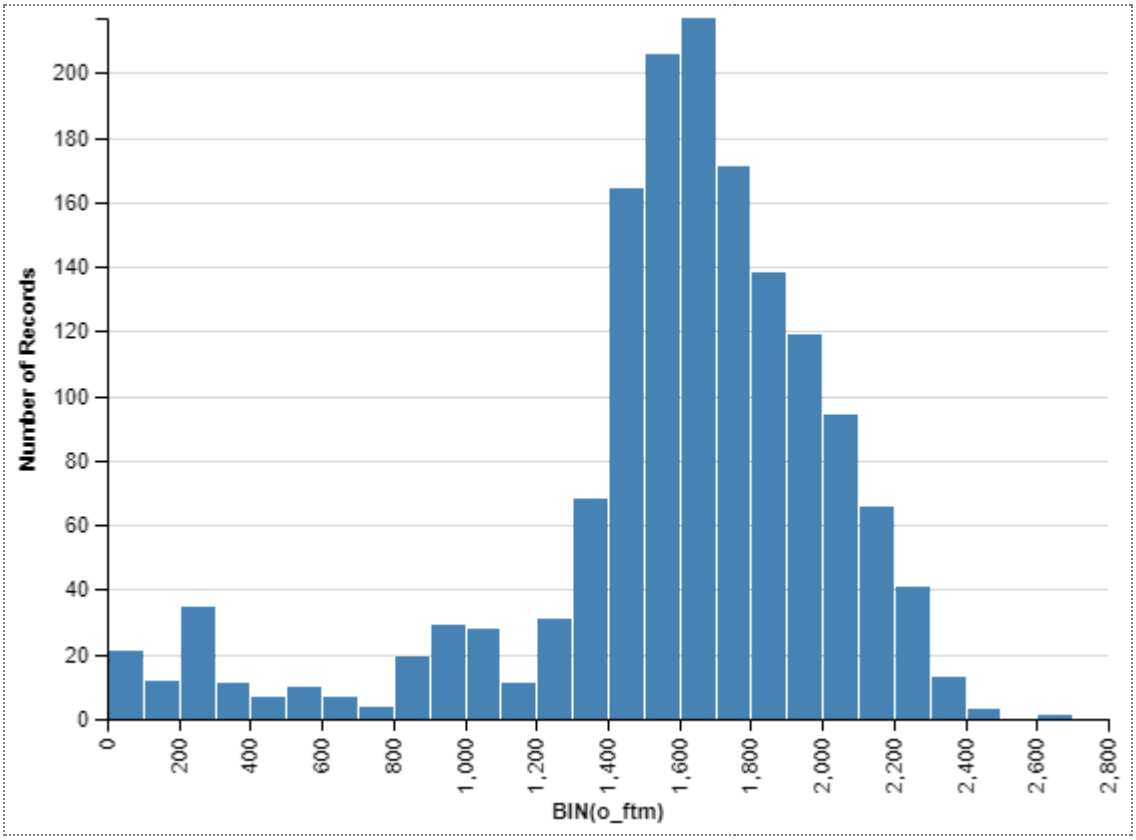


Export as PNG View Source Open in Vega Editor

In [909]:
```
# checking for the missing data in o_ftm column in teams_df:
len(teams_df[(teams_df.o_ftm.isnull() == True) | (teams_df.o_ftm == 0)])
```

Out[909]: 10

In [910]:
```
# Checking for the distribution of o_ftm column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_ftm.notnull()) & (teams_df.o_ftm != 
alt.Chart(teams_df[(teams_df.o_ftm.notnull()) & (teams_df.o_ftm != 0)]).mark_bar(
    x=alt.X('o_ftm', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

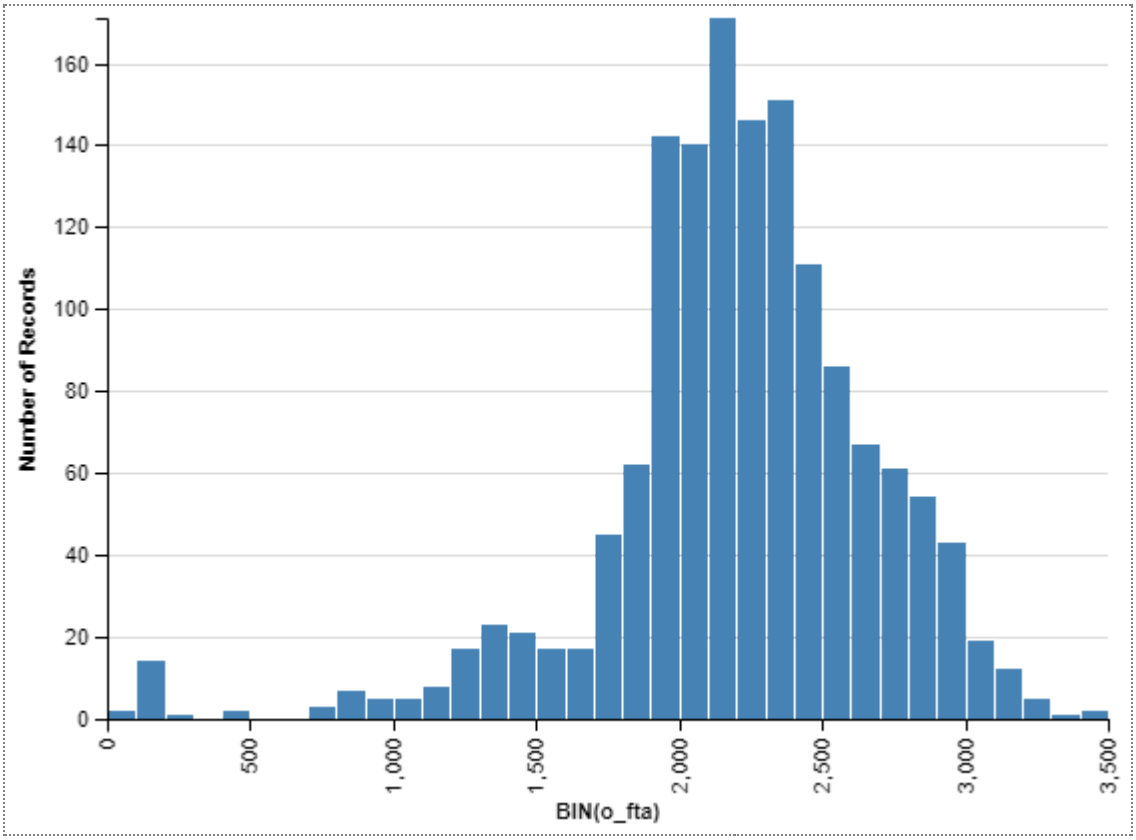


Export as PNG View Source Open in Vega Editor

In [911]:
```
# checking for the missing data in o_fta column in teams_df:
len(teams_df[(teams_df.o_fta.isnull() == True) | (teams_df.o_fta == 0)])
```

Out[911]: 76

In [912]:
```
# Checking for the distribution of o_fta column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_fta.notnull()) & (teams_df.o_fta !=
alt.Chart(teams_df[(teams_df.o_fta.notnull()) & (teams_df.o_fta != 0)]).mark_bar(
    x=alt.X('o_fta', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```



Export as PNG View Source Open in Vega Editor

In [913]:
```
# checking for the missing data in o_3pm column in teams_df:
len(teams_df[(teams_df.o_3pm.isnull() == True) | (teams_df.o_3pm == 0)])
```

Out[913]: 534

In [914]:
```
# Checking for the distribution of o_3pm column in teams_df:
print('The maximum value for the o_3pm column is {} and the minimum is {}.'.forma
                        teams_df.o_3pm.max(), teams_df.o_3pm.min()))
```

```
The maximum value for the o_3pm column is 841 and the minimum is 0.
```

In [915]:
```
# checking for the missing data in o_3pa column in teams_df:
len(teams_df[(teams_df.o_3pa.isnull() == True) | (teams_df.o_3pa == 0)])
```

Out[915]: 534

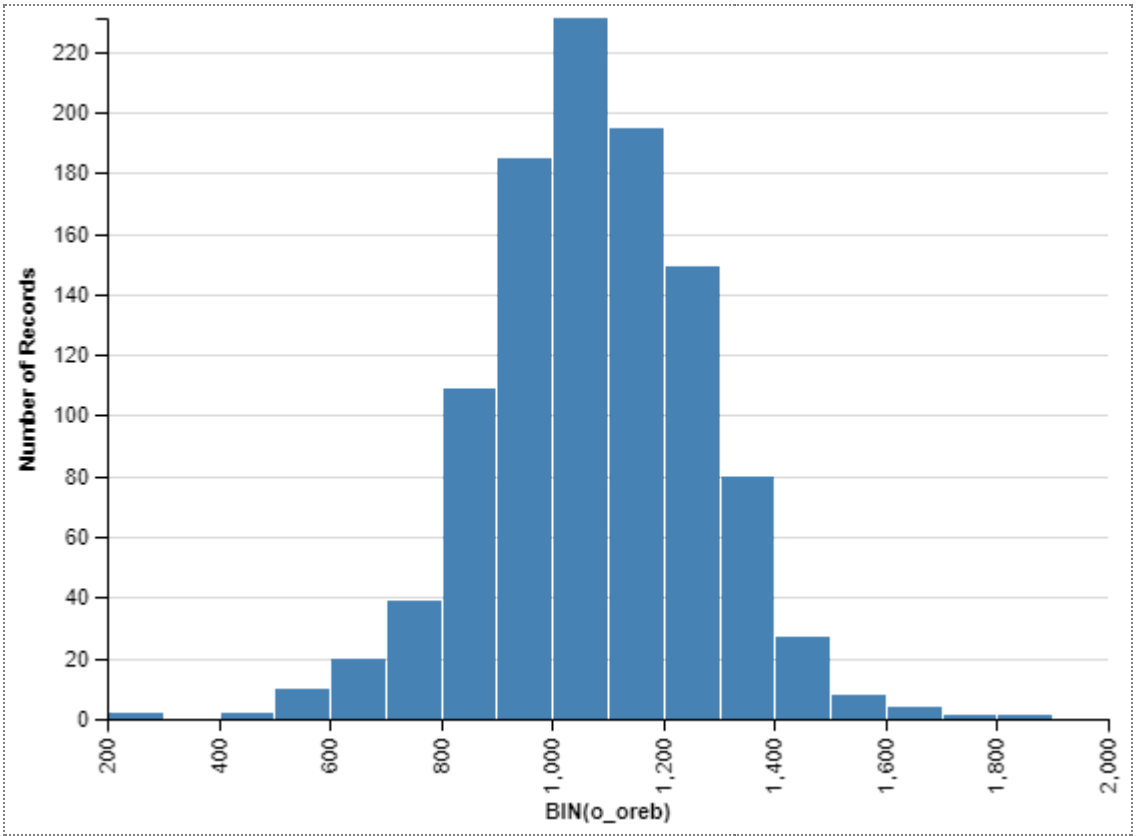
In [916]:
```
# Checking for the distribution of o_3pa column in teams_df:
print('The maximum value for the o_3pa column is {} and the minimum is {}.'.forma
                        teams_df.o_3pa.max(), teams_df.o_3pa.min()))
```

```
The maximum value for the o_3pa column is 2283 and the minimum is 0.
```

In [917]:
```
# checking for the missing data in o_oreb column in teams_df:
len(teams_df[(teams_df.o_oreb.isnull() == True) | (teams_df.o_oreb == 0)])
```

Out[917]: 473

In [918]:
```
# Checking for the distribution of o_oreb column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_oreb.notnull()) & (teams_df.o_oreb
alt.Chart(teams_df[(teams_df.o_oreb.notnull()) & (teams_df.o_oreb != 0)]).mark_ba
    x=alt.X('o_oreb', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

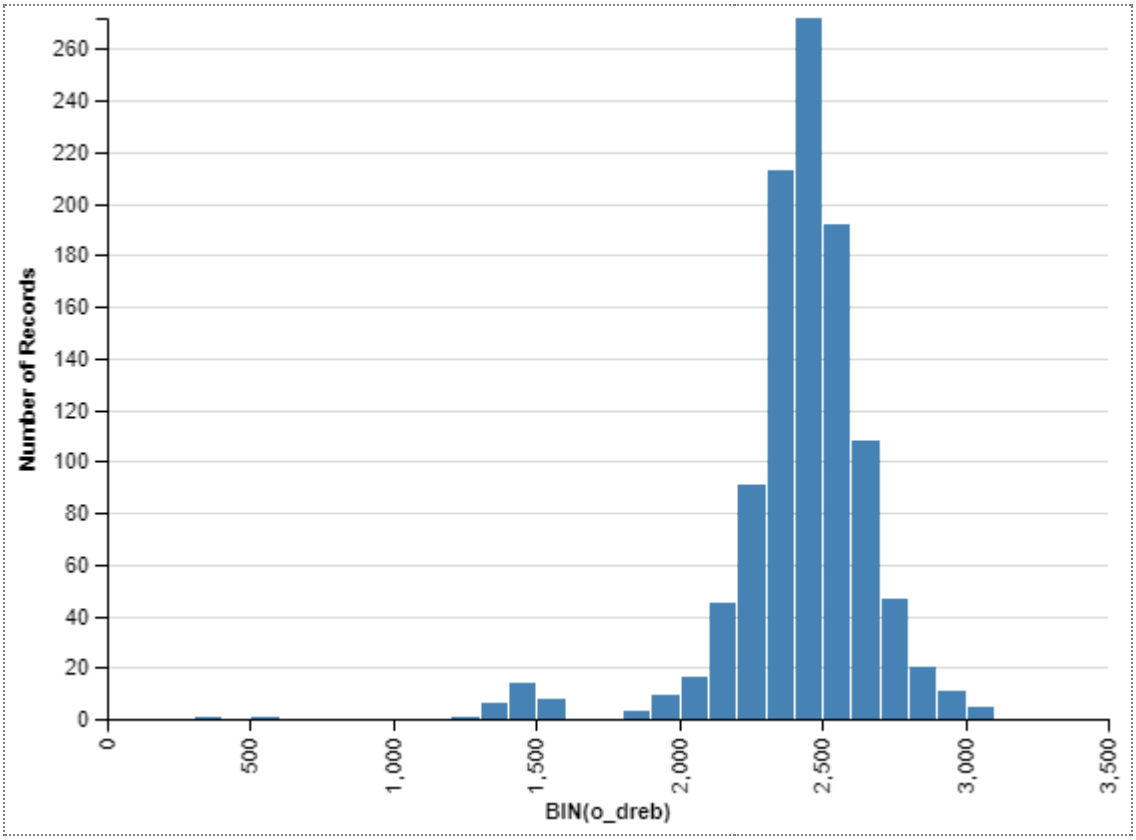


Export as PNG View Source Open in Vega Editor

In [919]:
```
# checking for the missing data in o_dreb column in teams_df:
len(teams_df[(teams_df.o_dreb.isnull() == True) | (teams_df.o_dreb == 0)])
```

Out[919]: 473

In [920]:
```
# Checking for the distribution of o_dreb column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_dreb.notnull()) & (teams_df.o_dreb
alt.Chart(teams_df[(teams_df.o_dreb.notnull()) & (teams_df.o_dreb != 0)]).mark_ba
    x=alt.X('o_dreb', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

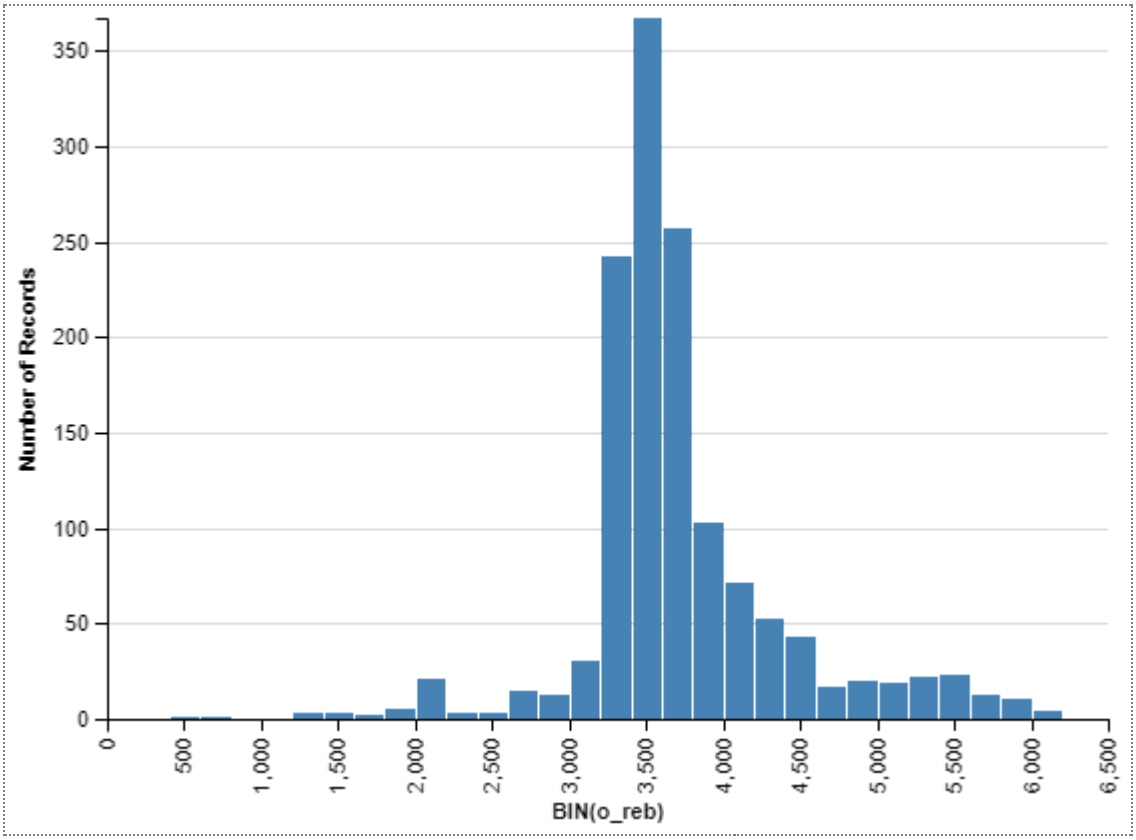


Export as PNG View Source Open in Vega Editor

In [921]:
```
# checking for the missing data in o_reb column in teams_df:
len(teams_df[(teams_df.o_reb.isnull() == True) | (teams_df.o_reb == 0)])
```

Out[921]: 173

In [922]:
```
# Checking for the distribution of o_reb column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_reb.notnull()) & (teams_df.o_reb !=
alt.Chart(teams_df[(teams_df.o_reb.notnull()) & (teams_df.o_reb != 0)]).mark_bar(
    x=alt.X('o_reb', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

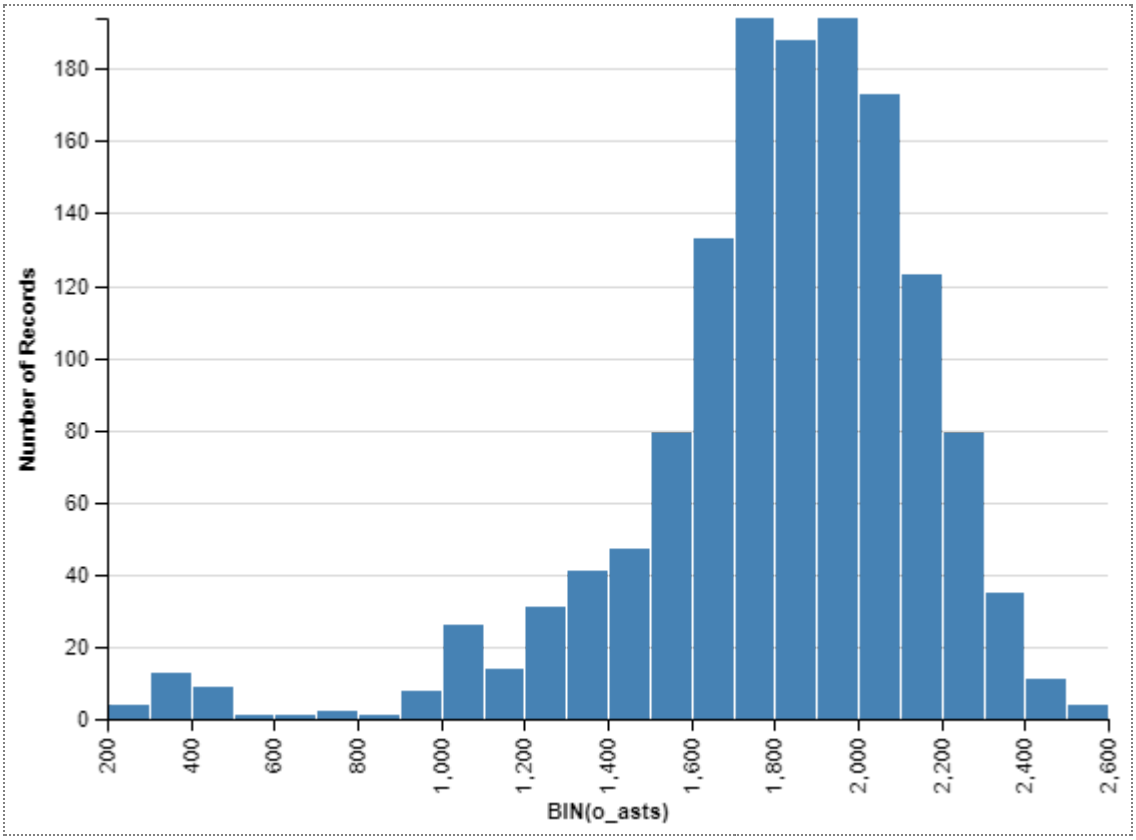


Export as PNG View Source Open in Vega Editor

In [923]:
```
# checking for the missing data in o_asts column in teams_df:
len(teams_df[(teams_df.o_asts.isnull() == True) | (teams_df.o_asts == 0)])
```

Out[923]: 125

In [924]:
```
# Checking for the distribution of o_asts column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_asts.notnull()) & (teams_df.o_asts
alt.Chart(teams_df[(teams_df.o_asts.notnull()) & (teams_df.o_asts != 0)]).mark_ba
    x=alt.X('o_asts', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

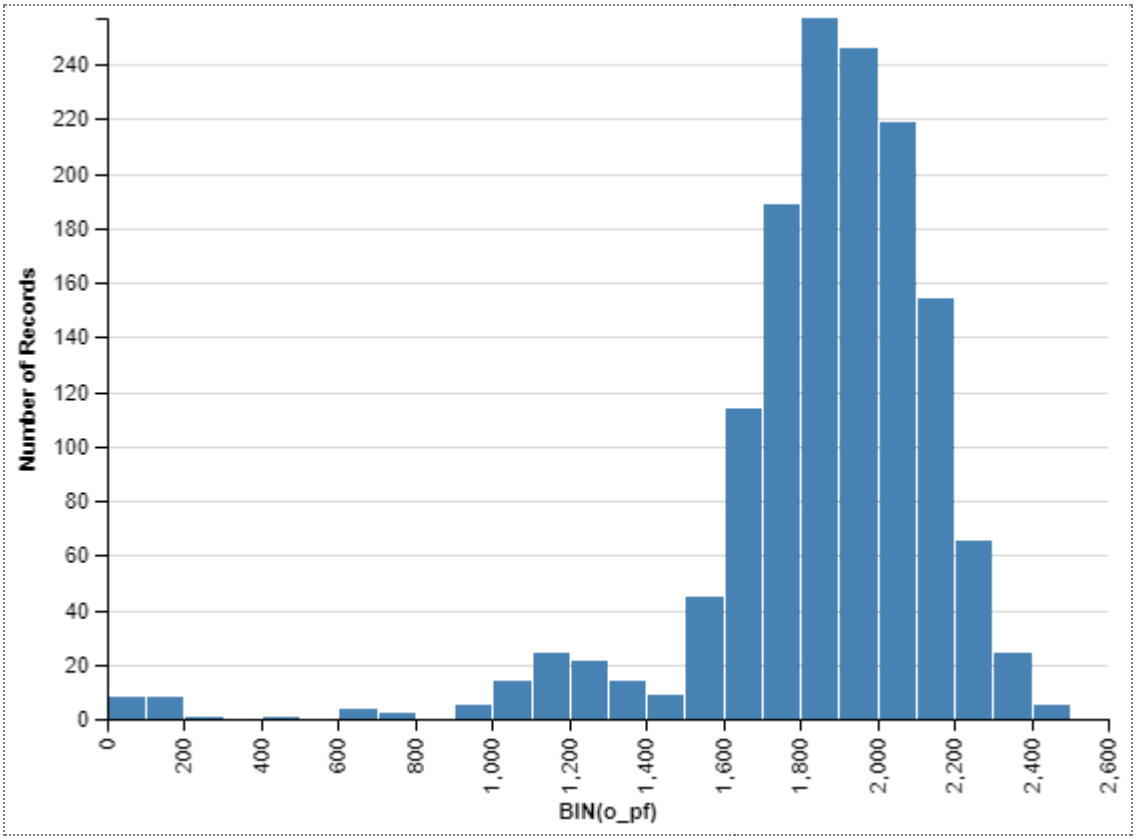


Export as PNG View Source Open in Vega Editor

In [925]:
```
# checking for the missing data in o_pf column in teams_df:
len(teams_df[(teams_df.o_pf.isnull() == True) | (teams_df.o_pf == 0)])
```

Out[925]: 107

In [926]:
```
# Checking for the distribution of o_pf column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_pf.notnull()) & (teams_df.o_pf != 0
alt.Chart(teams_df[(teams_df.o_pf.notnull()) & (teams_df.o_pf != 0)]).mark_bar().
    x=alt.X('o_pf', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

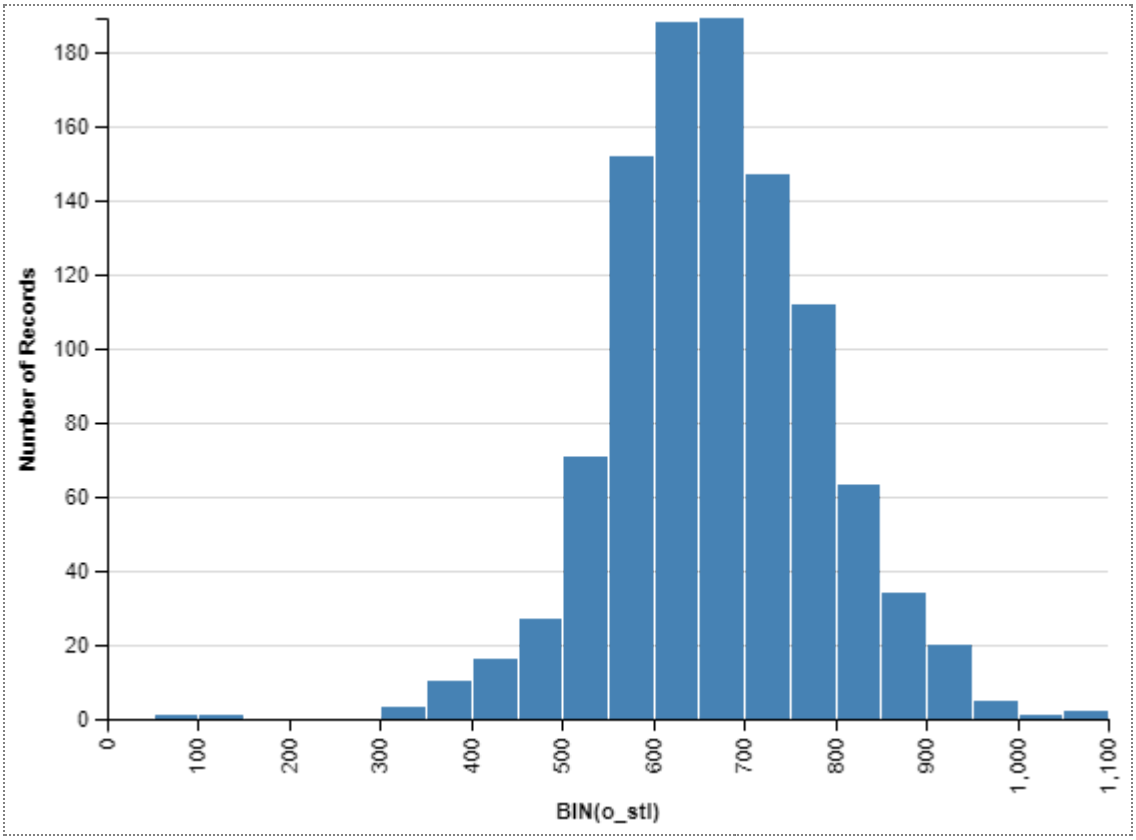


Export as PNG View Source Open in Vega Editor

In [927]:
```
# checking for the missing data in o_stl column in teams_df:
len(teams_df[(teams_df.o_stl.isnull() == True) | (teams_df.o_stl == 0)])
```

Out[927]: 494

In [928]:
```
# Checking for the distribution of o_stl column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_stl.notnull()) & (teams_df.o_stl !=
alt.Chart(teams_df[(teams_df.o_stl.notnull()) & (teams_df.o_stl != 0)]).mark_bar(
    x=alt.X('o_stl', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

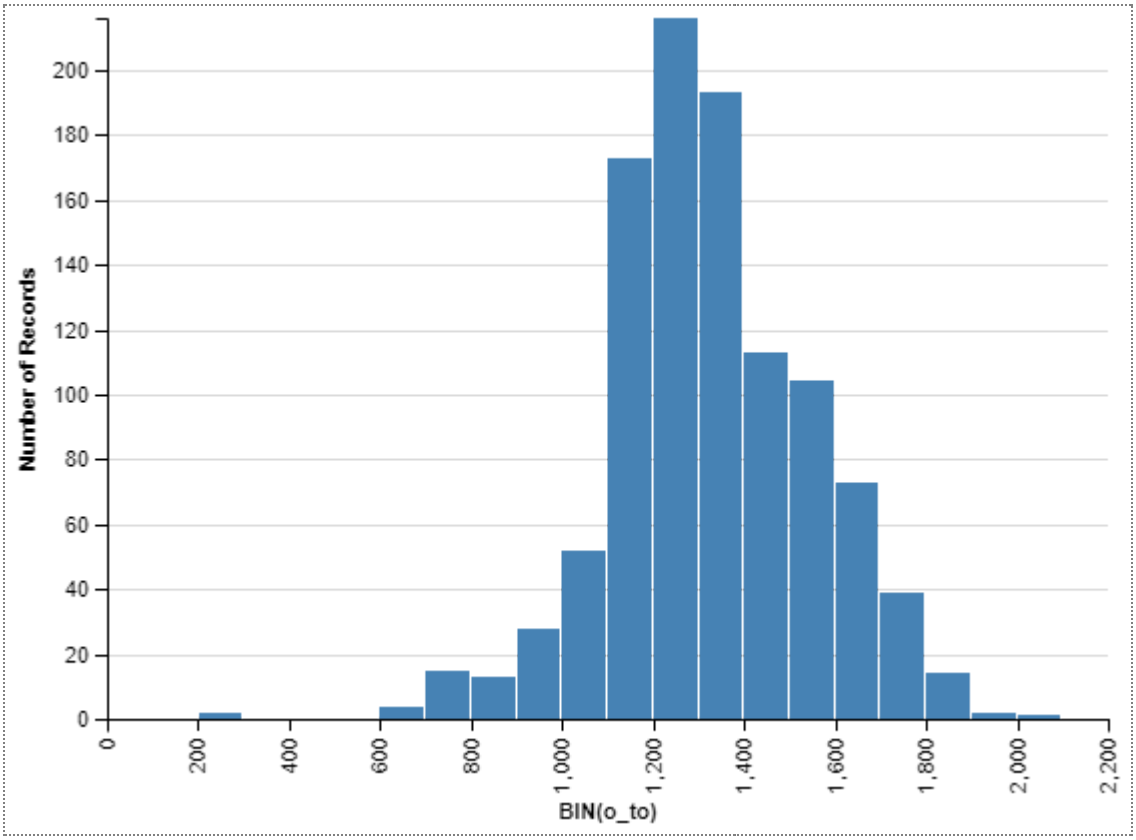


Export as PNG View Source Open in Vega Editor

In [929]:
```
# checking for the missing data in o_to column in teams_df:
len(teams_df[(teams_df.o_to.isnull() == True) | (teams_df.o_to == 0)])
```

Out[929]: 429

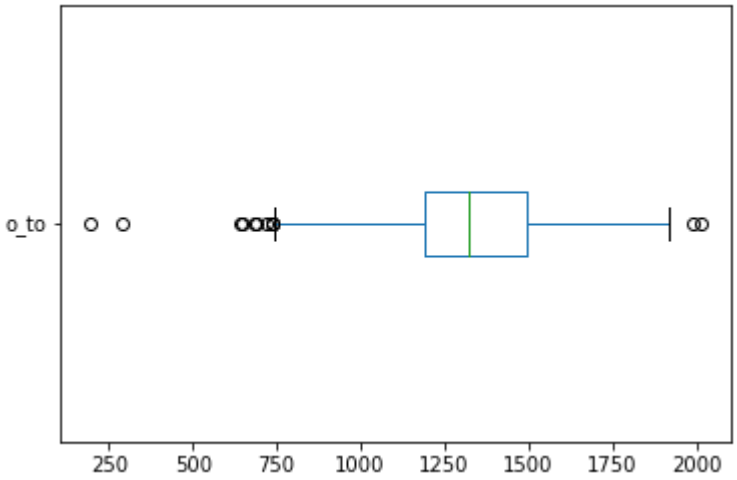
In [930]:
```
# Checking for the distribution of o_to column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_to.notnull()) & (teams_df.o_to != 0
alt.Chart(teams_df[(teams_df.o_stl.notnull()) & (teams_df.o_stl != 0)]).mark_bar(
    x=alt.X('o_to', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```



Export as PNG View Source Open in Vega Editor

In [931]:
```
teams_df.o_to[(teams_df.o_to.notnull()) & (teams_df.o_to != 0)].plot.box(vert=Fal
```

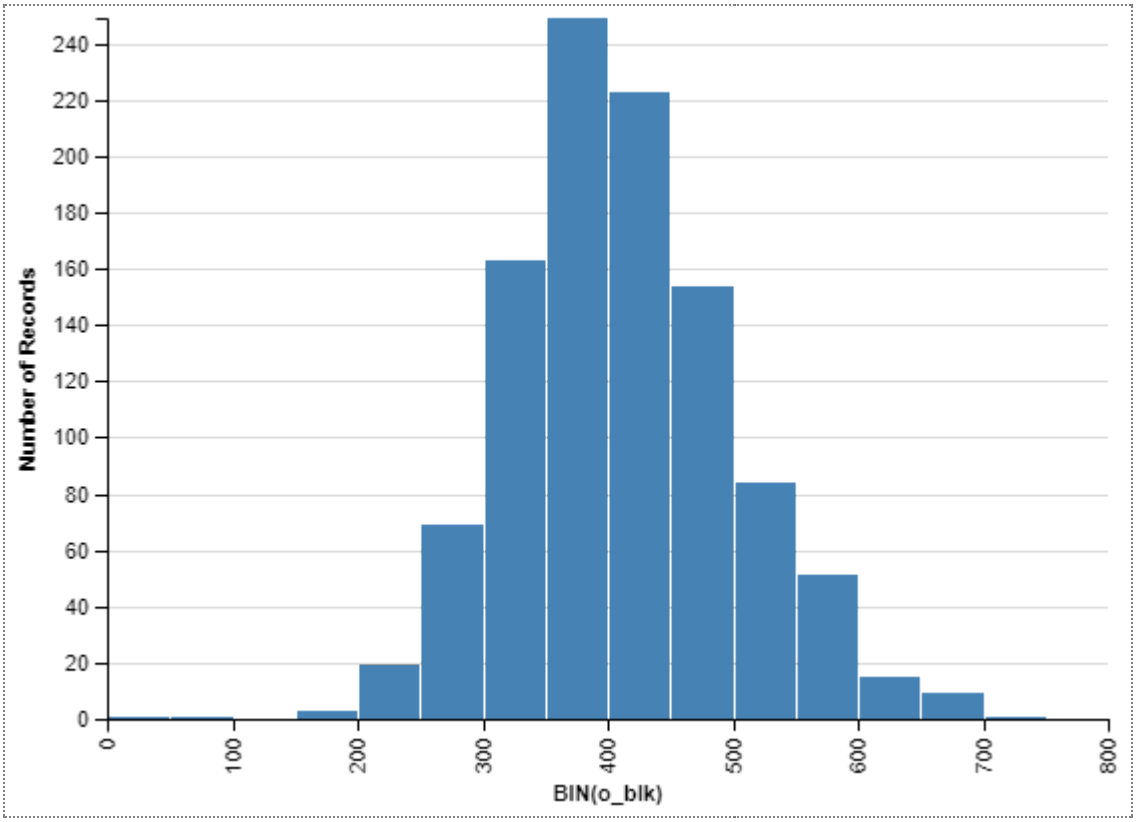
Out[931]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x14b316570b8>
```



In [932]:
```
# checking for the missing data in o_blk column in teams_df:
len(teams_df[(teams_df.o_blk.isnull() == True) | (teams_df.o_blk == 0)])
```

Out[932]:
```
494
```

In [933]:
```
# Checking for the distribution of o_blk column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_blk.notnull()) & (teams_df.o_blk !=
alt.Chart(teams_df[(teams_df.o_blk.notnull()) & (teams_df.o_blk != 0)]).mark_bar(
    x=alt.X('o_blk', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

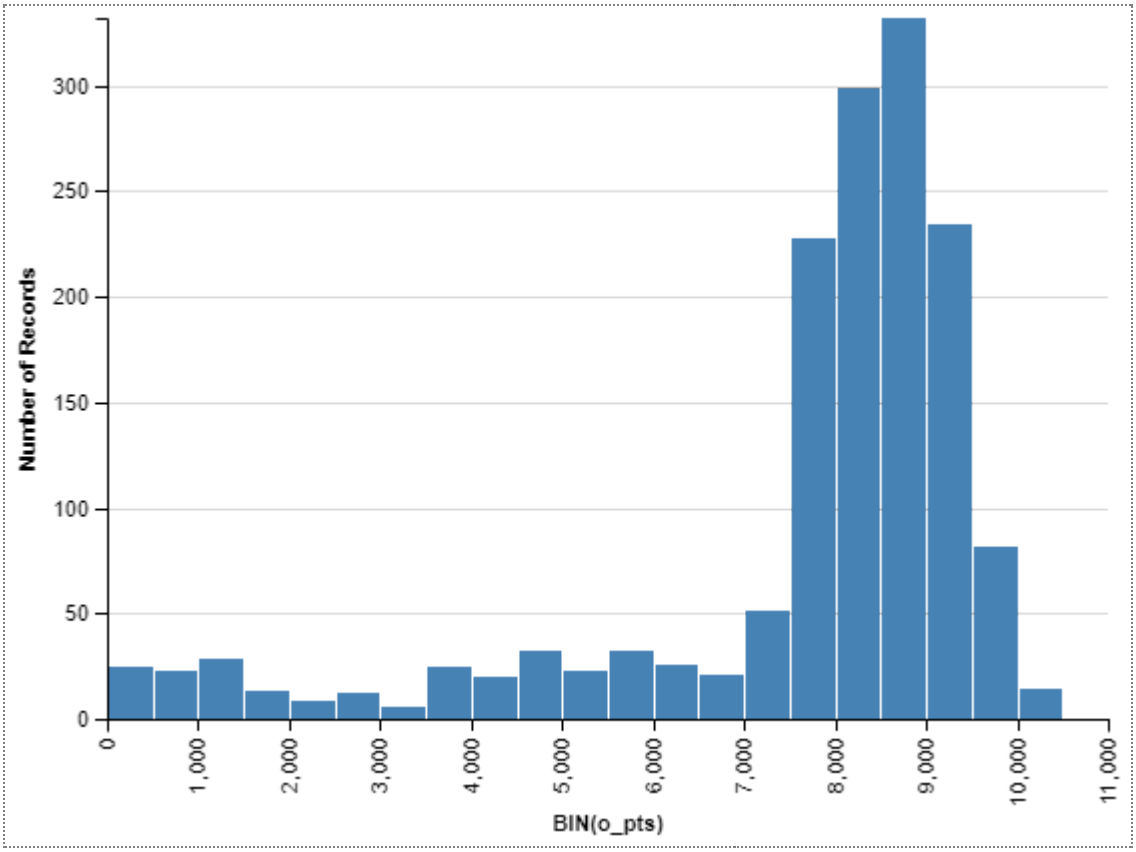


Export as PNG View Source Open in Vega Editor

In [934]:
```
# checking for the missing data in o_pts column in teams_df:
len(teams_df[(teams_df.o_pts.isnull() == True) | (teams_df.o_pts == 0)])
```

Out[934]: 1

In [935]:
```
# Checking for the distribution of o_pts column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_pts.notnull()) & (teams_df.o_pts !=
alt.Chart(teams_df[(teams_df.o_pts.notnull()) & (teams_df.o_pts != 0)]).mark_bar(
    x=alt.X('o_pts', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

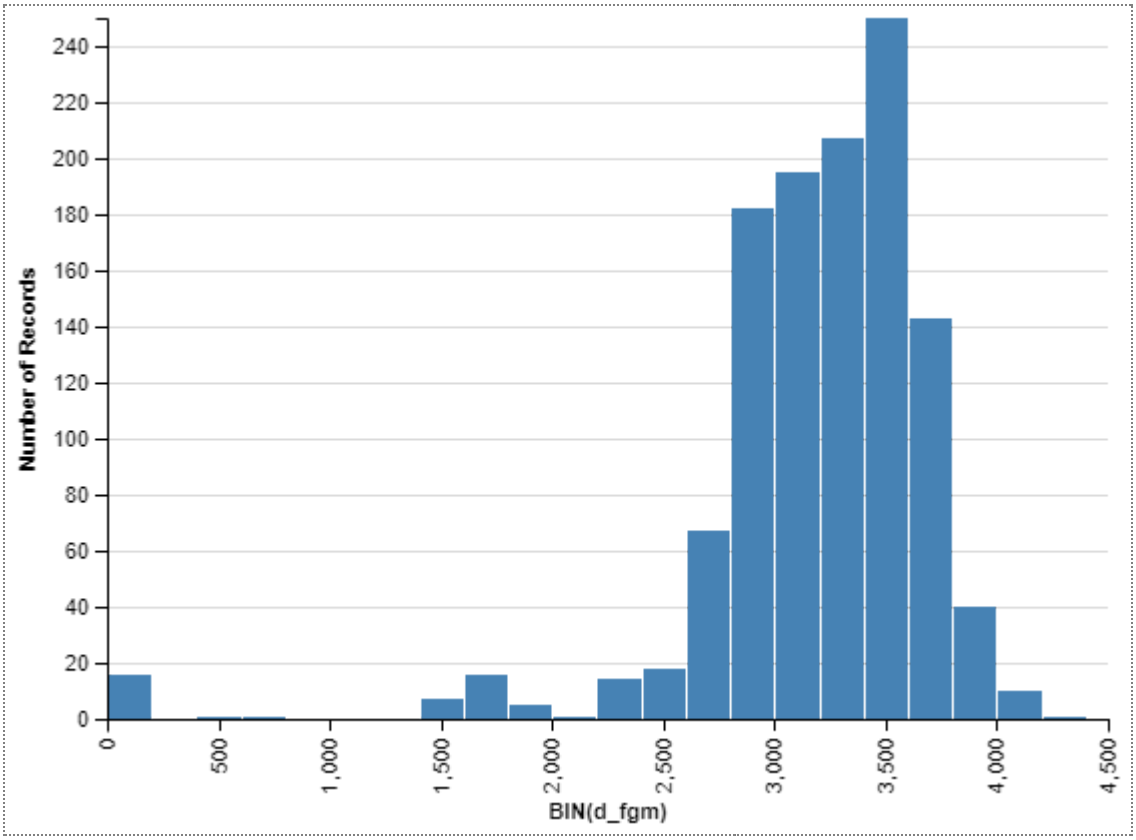


Export as PNG View Source Open in Vega Editor

In [936]:
```
# checking for the missing data in d_fgm column in teams_df:
len(teams_df[(teams_df.d_fgm.isnull() == True) | (teams_df.d_fgm == 0)])
```

Out[936]: 362

In [937]:
```
# Checking for the distribution of d_fgm column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_fgm.notnull()) & (teams_df.d_fgm !=
alt.Chart(teams_df[(teams_df.d_fgm.notnull()) & (teams_df.d_fgm != 0)]).mark_bar(
    x=alt.X('d_fgm', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

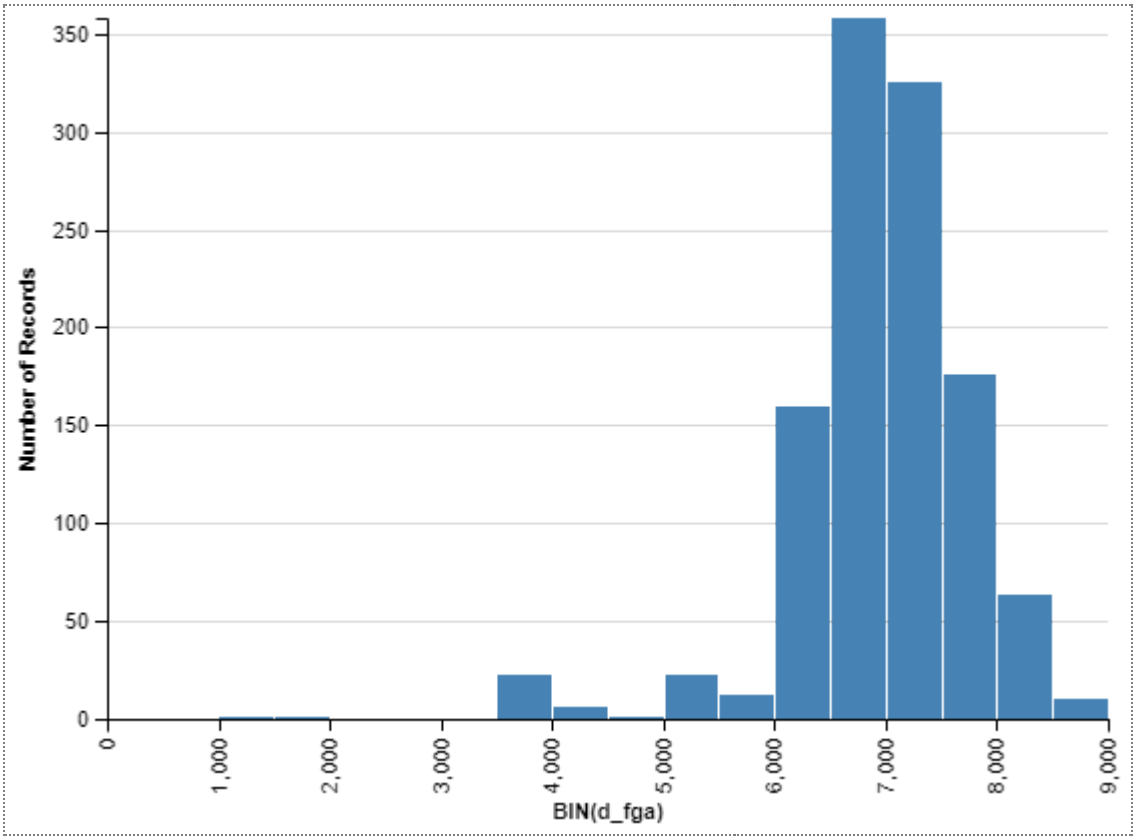


Export as PNG View Source Open in Vega Editor

In [938]:
```
# checking for the missing data in d_fga column in teams_df:
len(teams_df[(teams_df.d_fga.isnull() == True) | (teams_df.d_fga == 0)])
```

Out[938]: 378

In [939]:
```
# Checking for the distribution of d_fga column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_fga.notnull()) & (teams_df.d_fga !=
alt.Chart(teams_df[(teams_df.d_fga.notnull()) & (teams_df.d_fga != 0)]).mark_bar(
    x=alt.X('d_fga', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

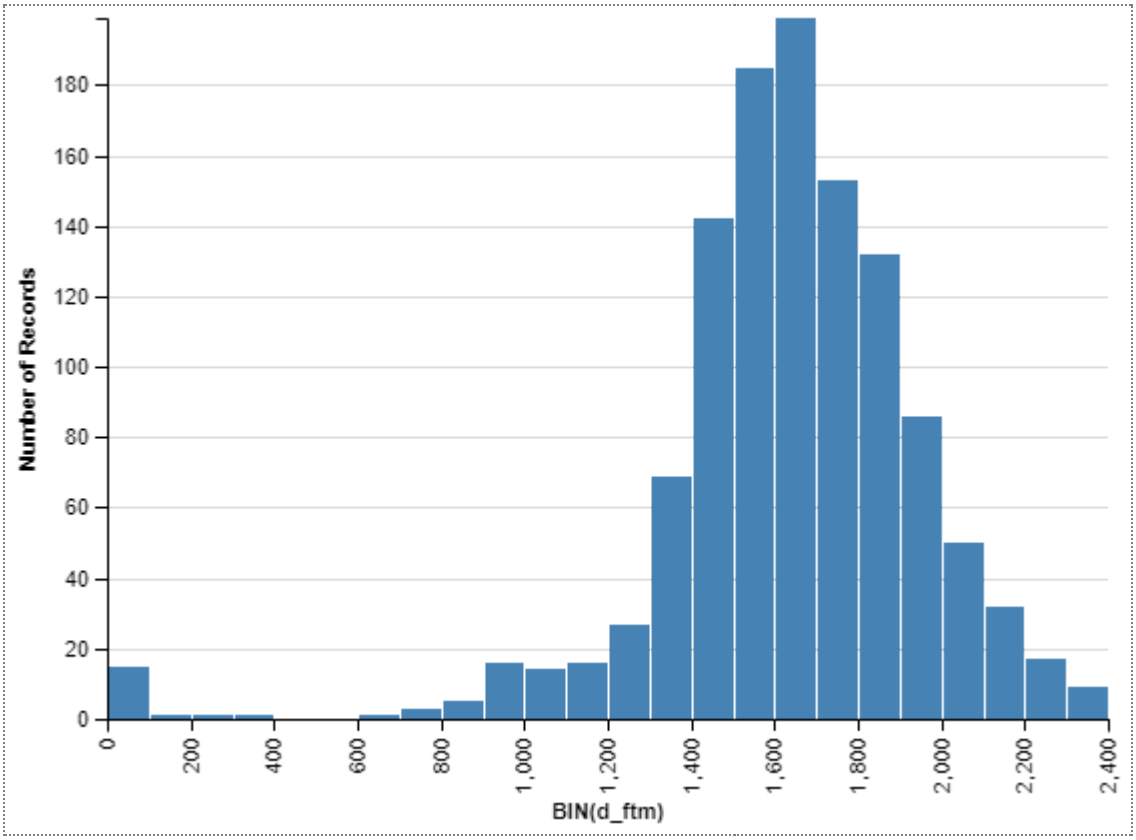


Export as PNG View Source Open in Vega Editor

In [940]:
```
# checking for the missing data in d_ftm column in teams_df:
len(teams_df[(teams_df.d_ftm.isnull() == True) | (teams_df.d_ftm == 0)])
```

Out[940]: 362

In [941]:
```
# Checking for the distribution of d_ftm column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_ftm.notnull()) & (teams_df.d_ftm !=
alt.Chart(teams_df[(teams_df.d_ftm.notnull()) & (teams_df.d_ftm != 0)]).mark_bar(
    x=alt.X('d_ftm', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

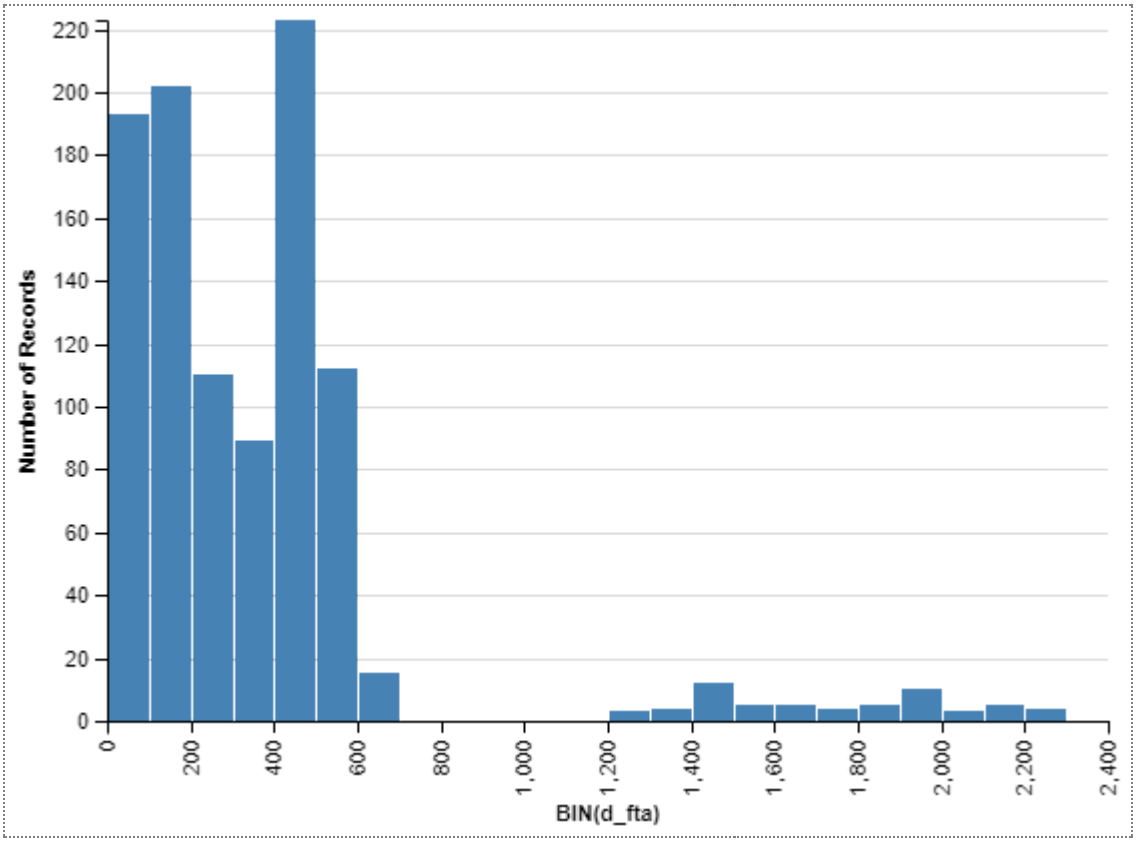


Export as PNG View Source Open in Vega Editor

In [942]:
```
# checking for the missing data in d_fta column in teams_df:
len(teams_df[(teams_df.d_fta.isnull() == True) | (teams_df.d_fta == 0)])
```

Out[942]: 532

In [943]:
```
# Checking for the distribution of d_fta column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_ftm.notnull()) & (teams_df.d_fta !=
alt.Chart(teams_df[(teams_df.d_fta.notnull()) & (teams_df.d_fta != 0)]).mark_bar(
    x=alt.X('d_fta', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```



Export as PNG View Source Open in Vega Editor

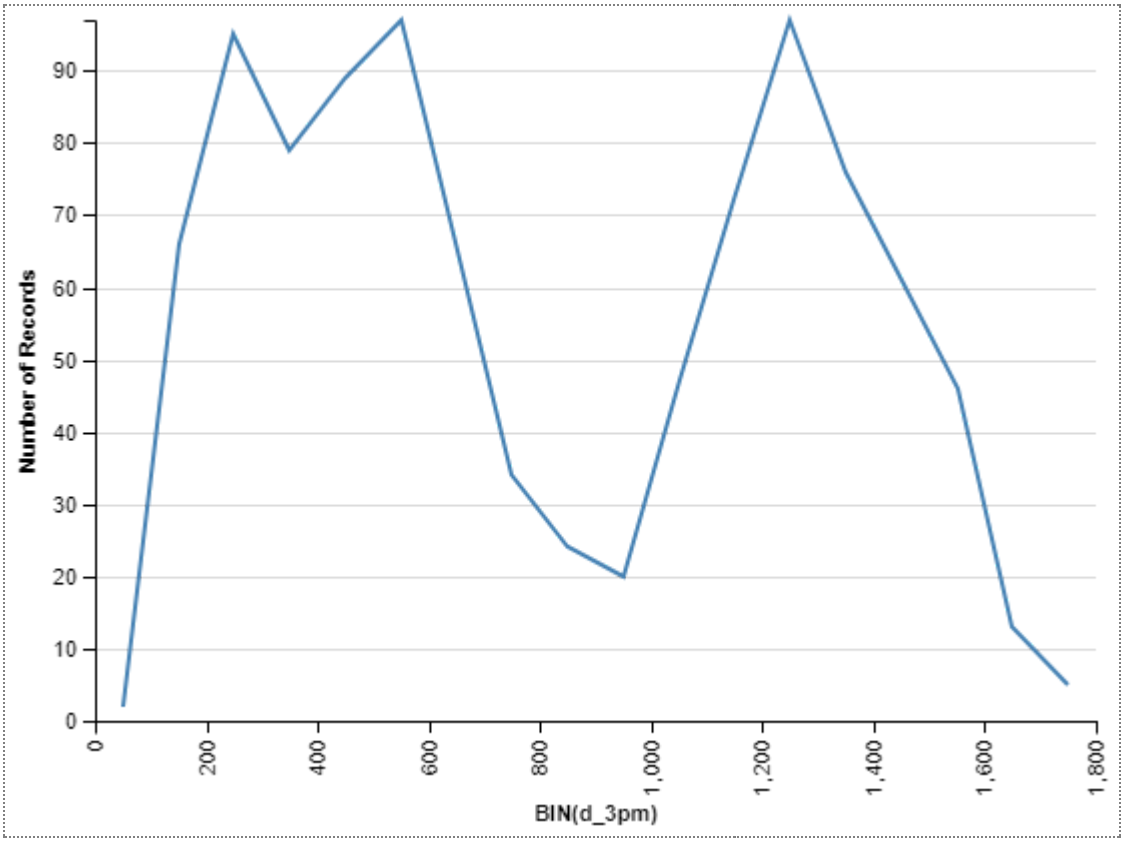
In [944]:
```
# Checking for the distribution of d_fta column in teams_df:
print('The maximum value for the d_fta column is {} and the minimum is {}.'.forma
                        teams_df.d_fta.max(), teams_df.d_fta[(teams_df.d_ftm.notn
```

```
The maximum value for the d_fta column is 2285 and the minimum is 12.
```

In [945]:
```
# checking for the missing data in d_3pm column in teams_df:
len(teams_df[(teams_df.d_3pm.isnull() == True) | (teams_df.d_3pm == 0)])
```

Out[945]: 548

In [946]:
```
# Checking for the distribution of d_3pm column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_3pm.notnull()) & (teams_df.d_3pm !=
alt.Chart(teams_df[(teams_df.d_3pm.notnull()) & (teams_df.d_3pm != 0)]).mark_line
    x=alt.X('d_3pm', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

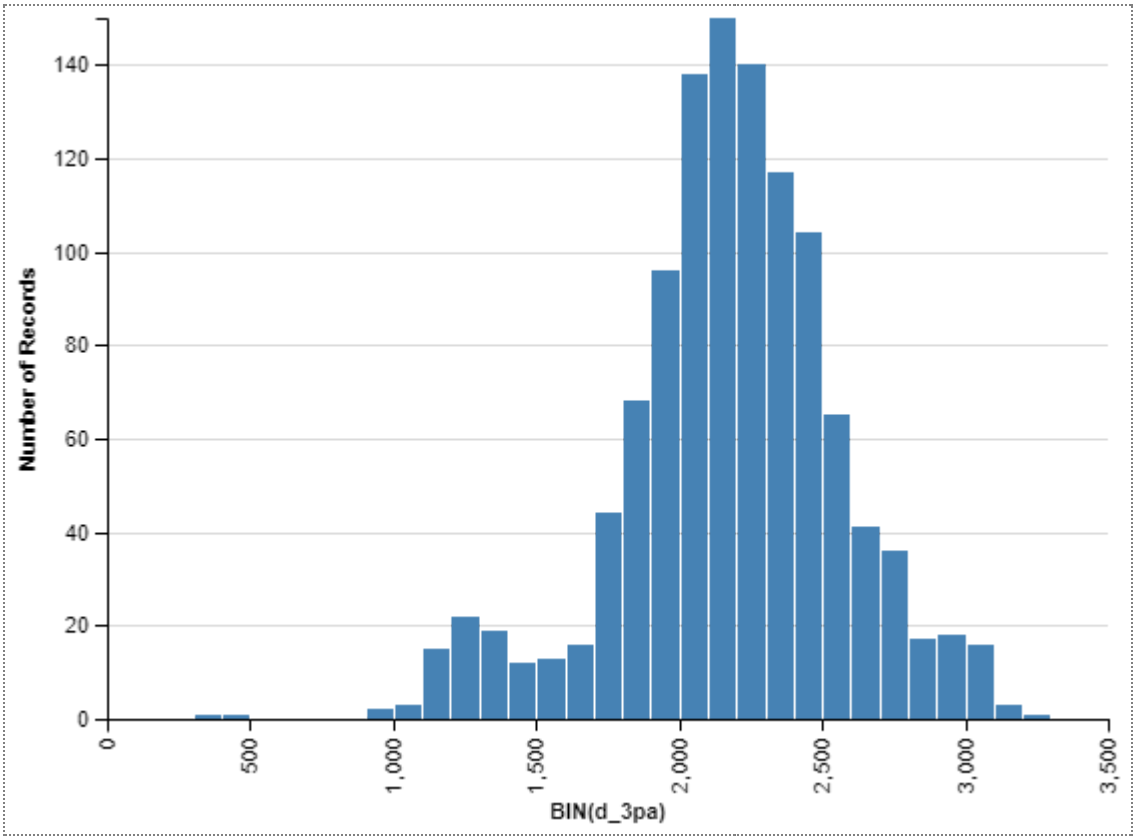


Export as PNG View Source Open in Vega Editor

In [947]:
```
# checking for the missing data in d_3pa column in teams_df:
len(teams_df[(teams_df.d_3pa.isnull() == True) | (teams_df.d_3pa == 0)])
```

Out[947]: 378

In [948]:
```
# Checking for the distribution of d_3pa column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_3pa.notnull()) & (teams_df.d_3pa !=
alt.Chart(teams_df[(teams_df.d_3pa.notnull()) & (teams_df.d_3pa != 0)]).mark_bar(
    x=alt.X('d_3pa', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

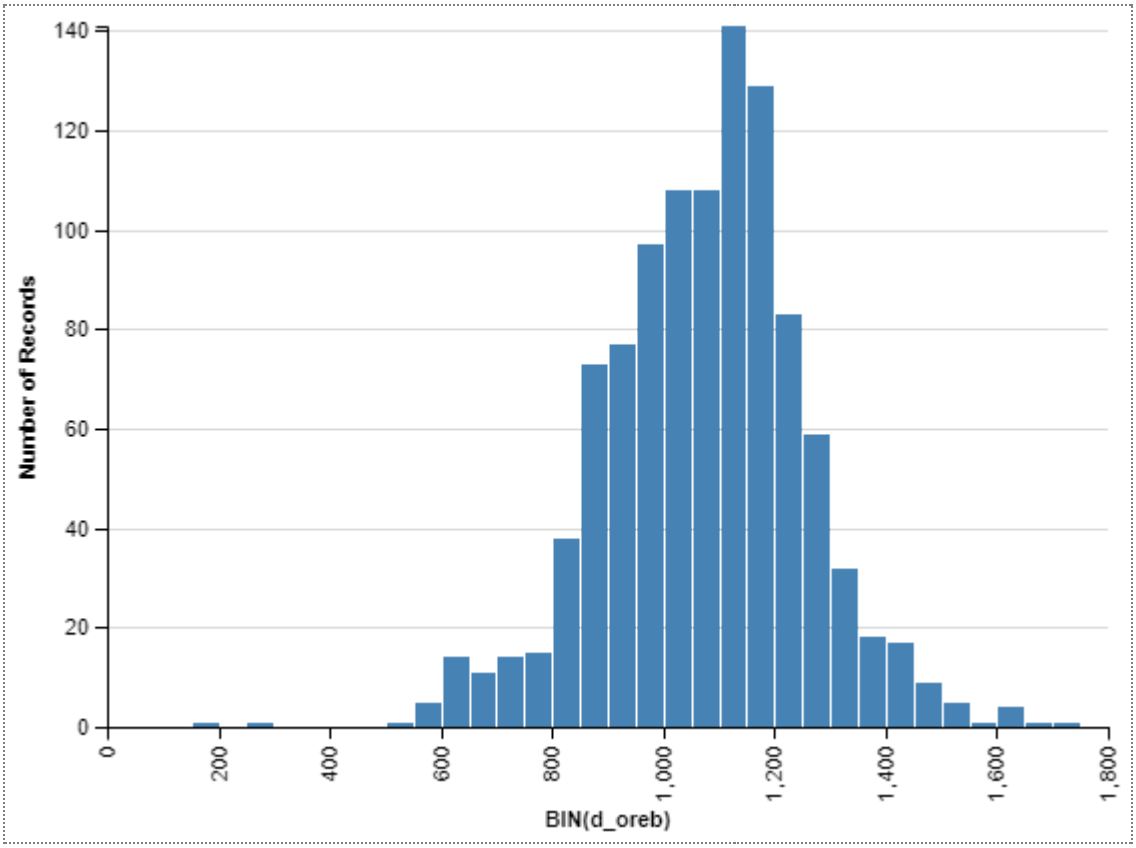


Export as PNG View Source Open in Vega Editor

In [949]:
```
# checking for the missing data in d_oreb column in teams_df:
len(teams_df[(teams_df.d_oreb.isnull() == True) | (teams_df.d_oreb == 0)])
```

Out[949]: 473

In [950]:
```
# Checking for the distribution of d_oreb column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_oreb.notnull()) & (teams_df.d_oreb
alt.Chart(teams_df[(teams_df.d_oreb.notnull()) & (teams_df.d_oreb != 0)]).mark_ba
    x=alt.X('d_oreb', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

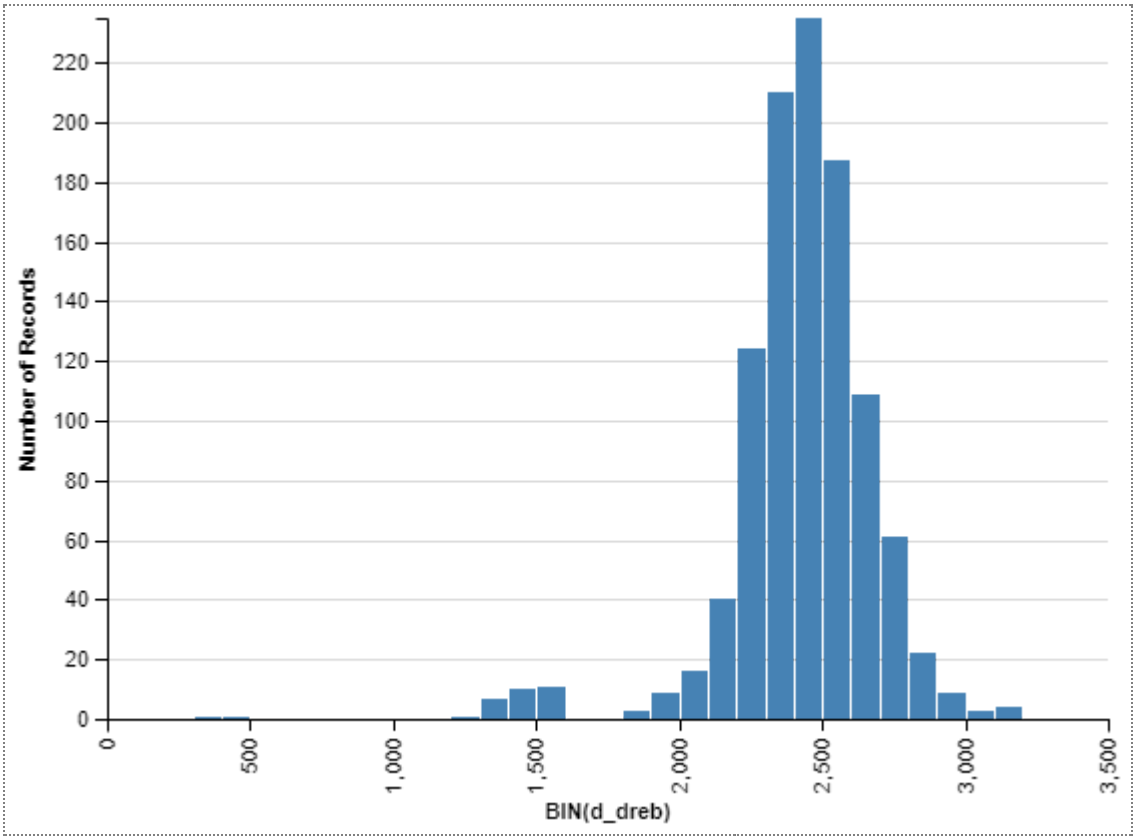


Export as PNG View Source Open in Vega Editor

In [951]:
```
# checking for the missing data in d_dreb column in teams_df:
len(teams_df[(teams_df.d_dreb.isnull() == True) | (teams_df.d_dreb == 0)])
```

Out[951]: 473

In [952]:
```
# Checking for the distribution of d_dreb column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_dreb.notnull()) & (teams_df.d_dreb
alt.Chart(teams_df[(teams_df.d_dreb.notnull()) & (teams_df.d_dreb != 0)]).mark_ba
    x=alt.X('d_dreb', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

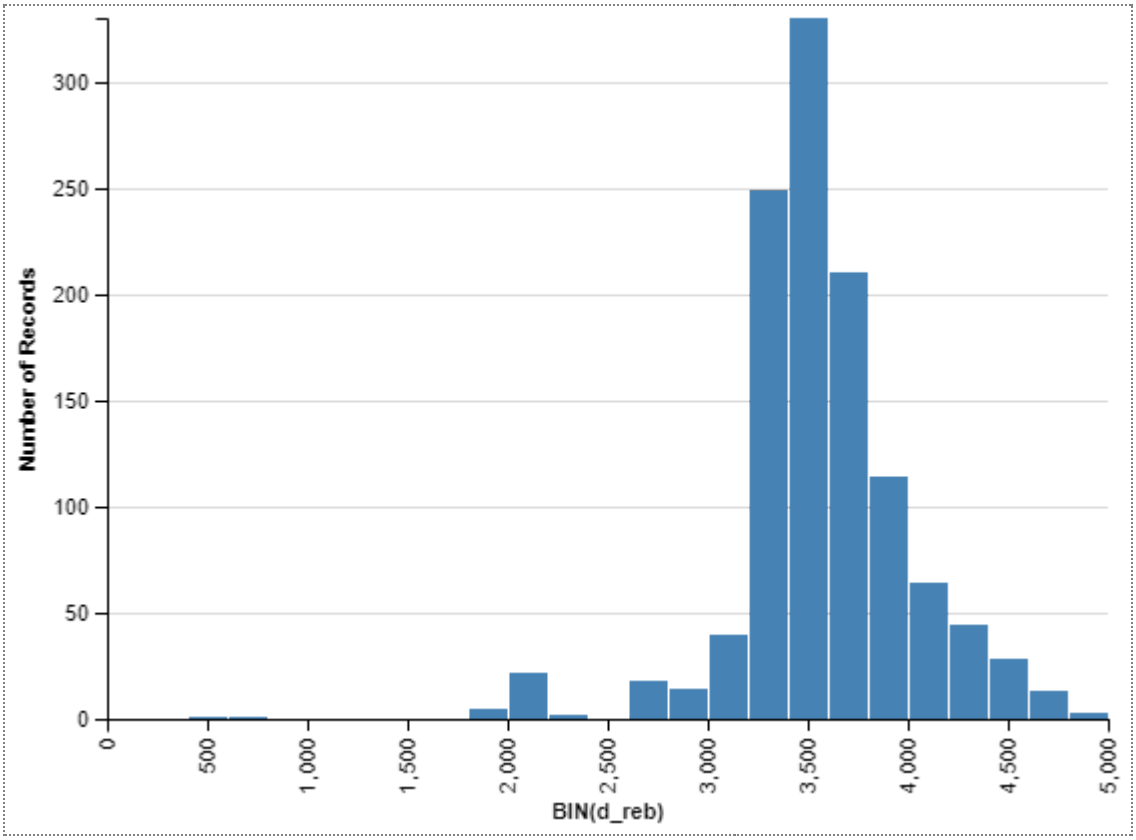


Export as PNG View Source Open in Vega Editor

In [953]:
```
# checking for the missing data in d_reb column in teams_df:
len(teams_df[(teams_df.d_reb.isnull() == True) | (teams_df.d_reb == 0)])
```

Out[953]: 378

In [954]:
```
# Checking for the distribution of d_reb column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_reb.notnull()) & (teams_df.d_reb !=
alt.Chart(teams_df[(teams_df.d_reb.notnull()) & (teams_df.d_reb != 0)]).mark_bar(
    x=alt.X('d_reb', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

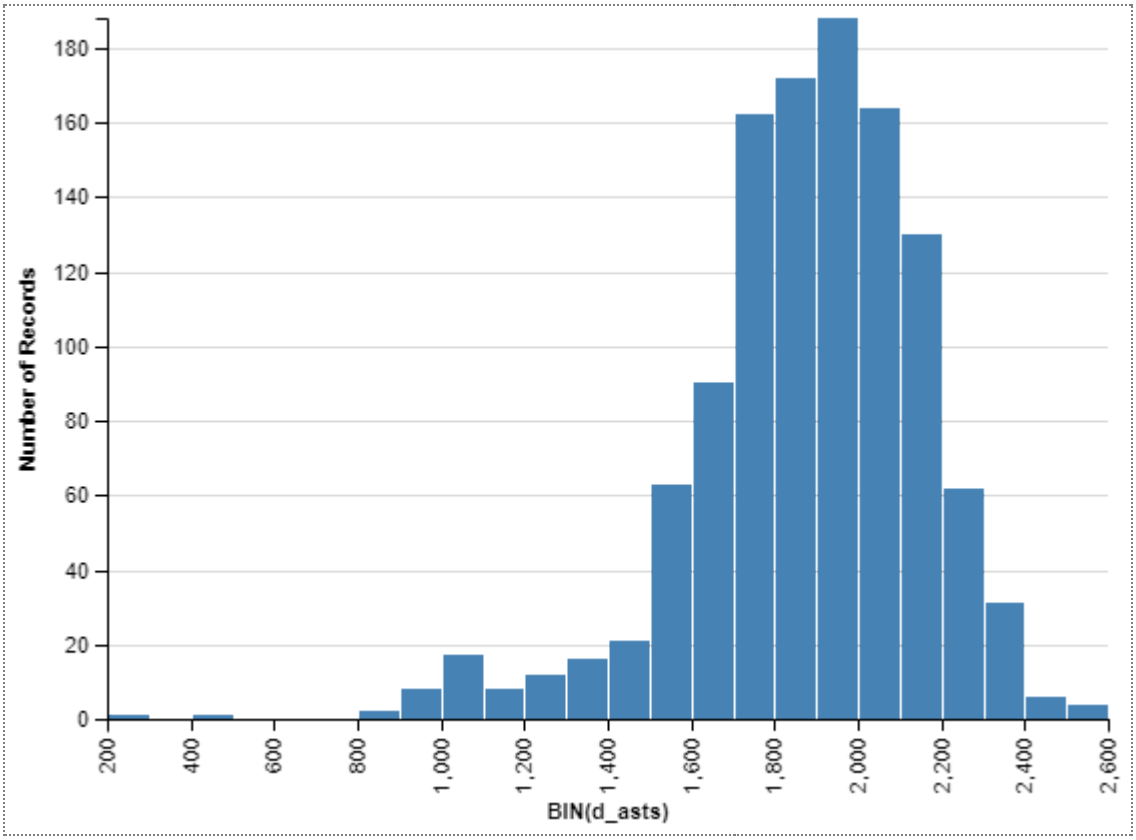


Export as PNG View Source Open in Vega Editor

In [955]:
```
# checking for the missing data in d_asts column in teams_df:
len(teams_df[(teams_df.d_asts.isnull() == True) | (teams_df.d_asts == 0)])
```

Out[955]: 378

In [956]:
```
# Checking for the distribution of d_asts column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_asts.notnull()) & (teams_df.d_asts
alt.Chart(teams_df[(teams_df.d_asts.notnull()) & (teams_df.d_asts != 0)]).mark_ba
    x=alt.X('d_asts', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

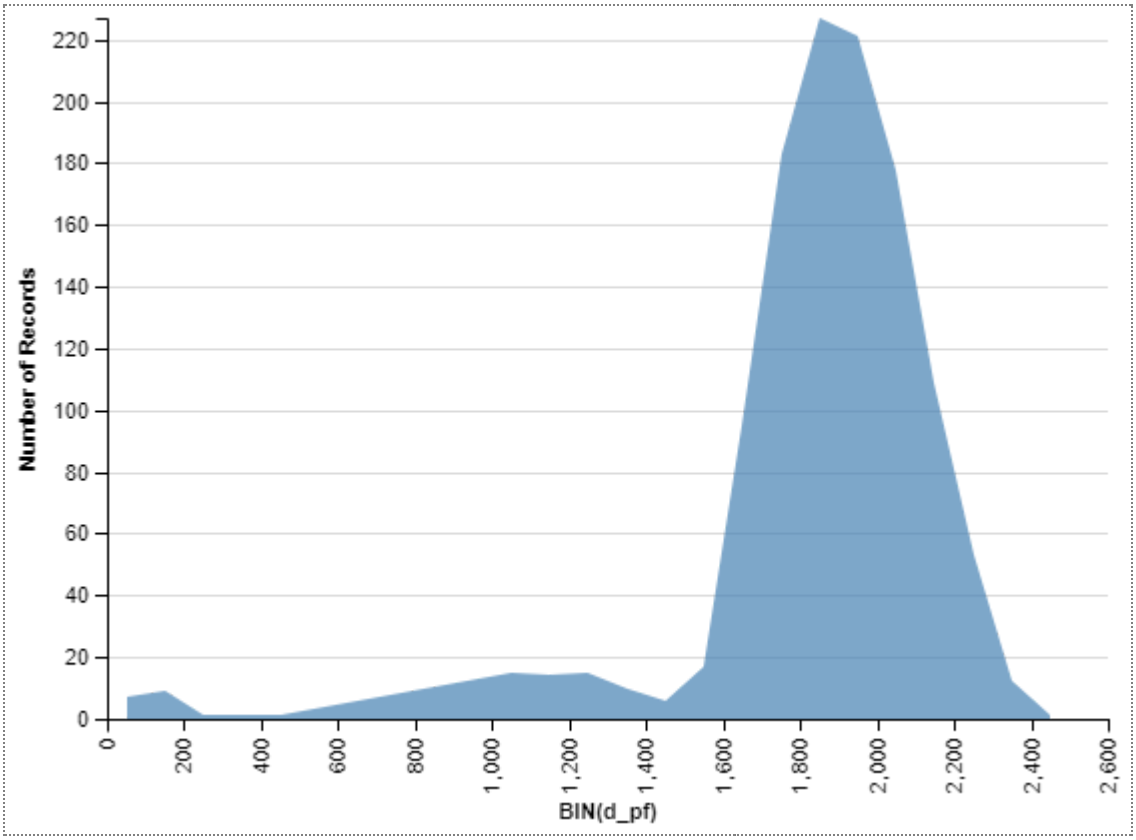


Export as PNG View Source Open in Vega Editor

In [957]:
```
# checking for the missing data in d_pf column in teams_df:
len(teams_df[(teams_df.d_pf.isnull() == True) | (teams_df.d_pf == 0)])
```

Out[957]: 362

In [958]:
```
# Checking for the distribution of d_pf column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_pf.notnull()) & (teams_df.d_pf != 0
alt.Chart(teams_df[(teams_df.d_pf.notnull()) & (teams_df.d_pf != 0)]).mark_area()
    x=alt.X('d_pf', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

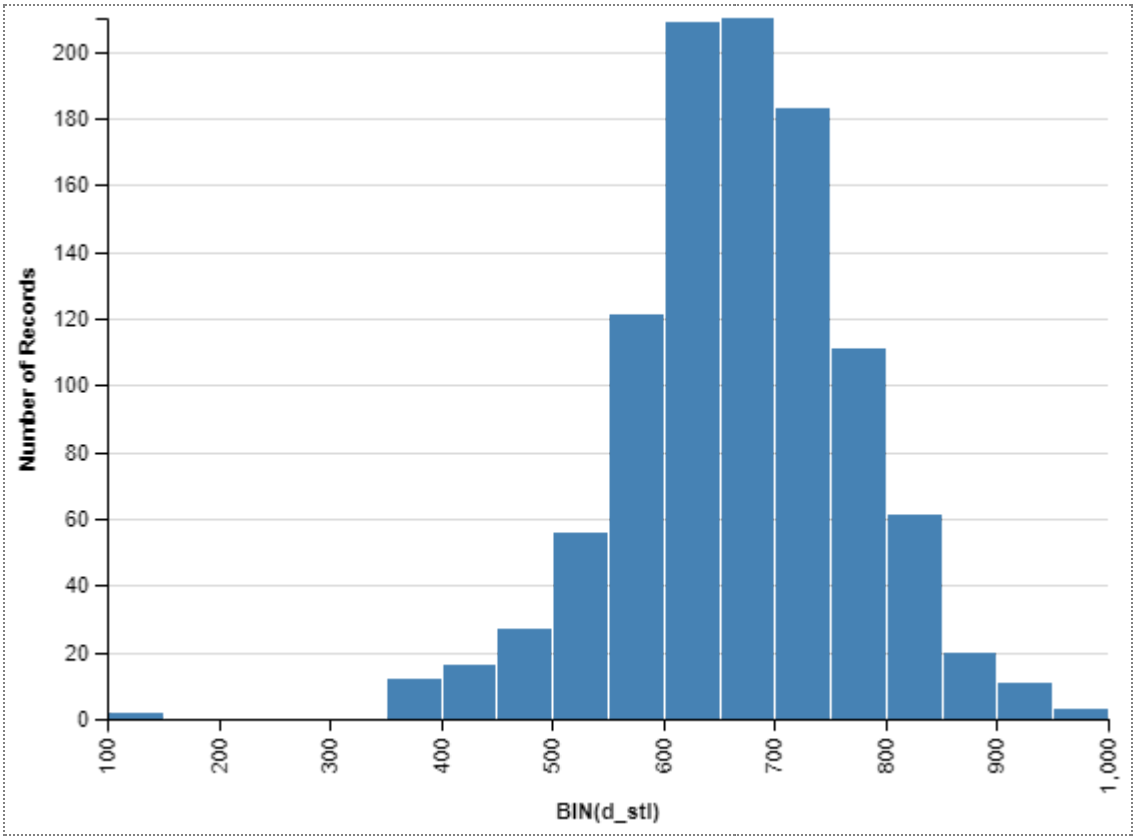


Export as PNG View Source Open in Vega Editor

In [959]:
```
# checking for the missing data in d_stl column in teams_df:
len(teams_df[(teams_df.d_stl.isnull() == True) | (teams_df.d_stl == 0)])
```

Out[959]: 494

In [960]:
```
# Checking for the distribution of d_stl column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_stl.notnull()) & (teams_df.d_stl !=
alt.Chart(teams_df[(teams_df.d_stl.notnull()) & (teams_df.d_stl != 0)]).mark_bar(
    x=alt.X('d_stl', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

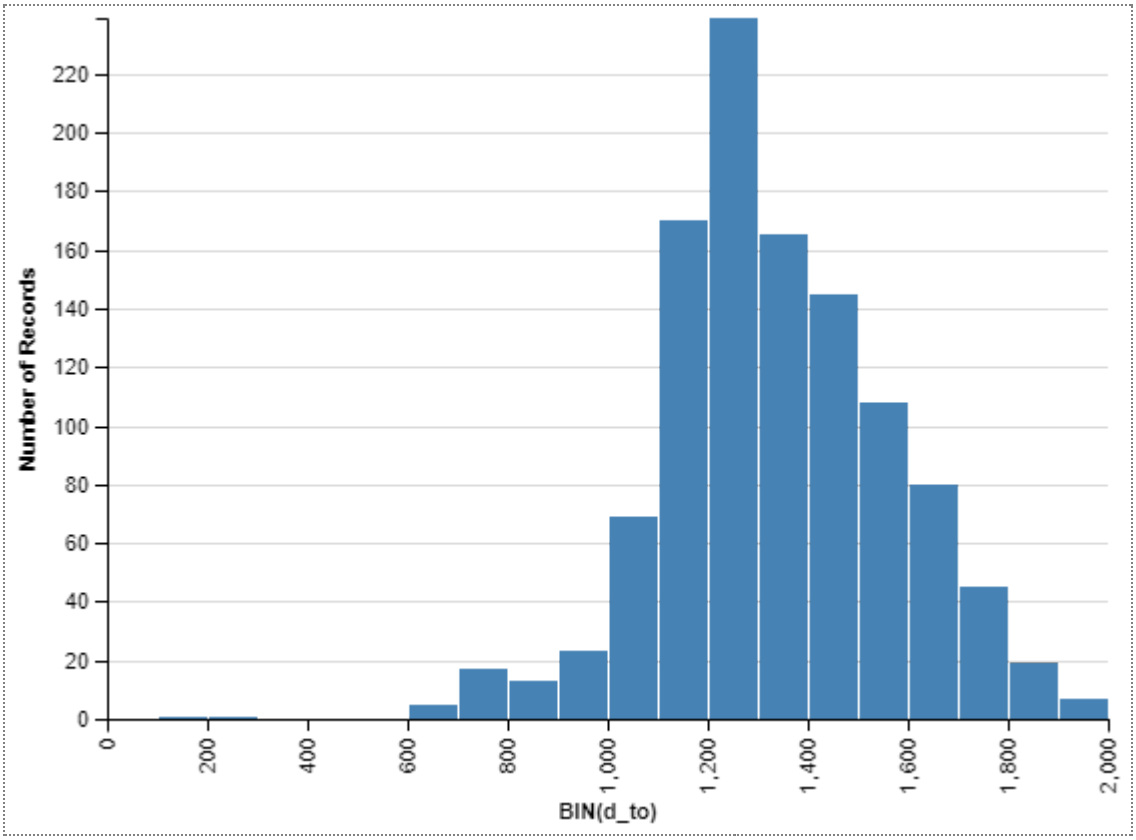


Export as PNG View Source Open in Vega Editor

In [961]:
```
# checking for the missing data in d_to column in teams_df:
len(teams_df[(teams_df.d_to.isnull() == True) | (teams_df.d_to == 0)])
```

Out[961]: 429

In [962]:
```
# Checking for the distribution of d_to column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_to.notnull()) & (teams_df.d_to != 0
alt.Chart(teams_df[(teams_df.d_to.notnull()) & (teams_df.d_to != 0)]).mark_bar().
    x=alt.X('d_to', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

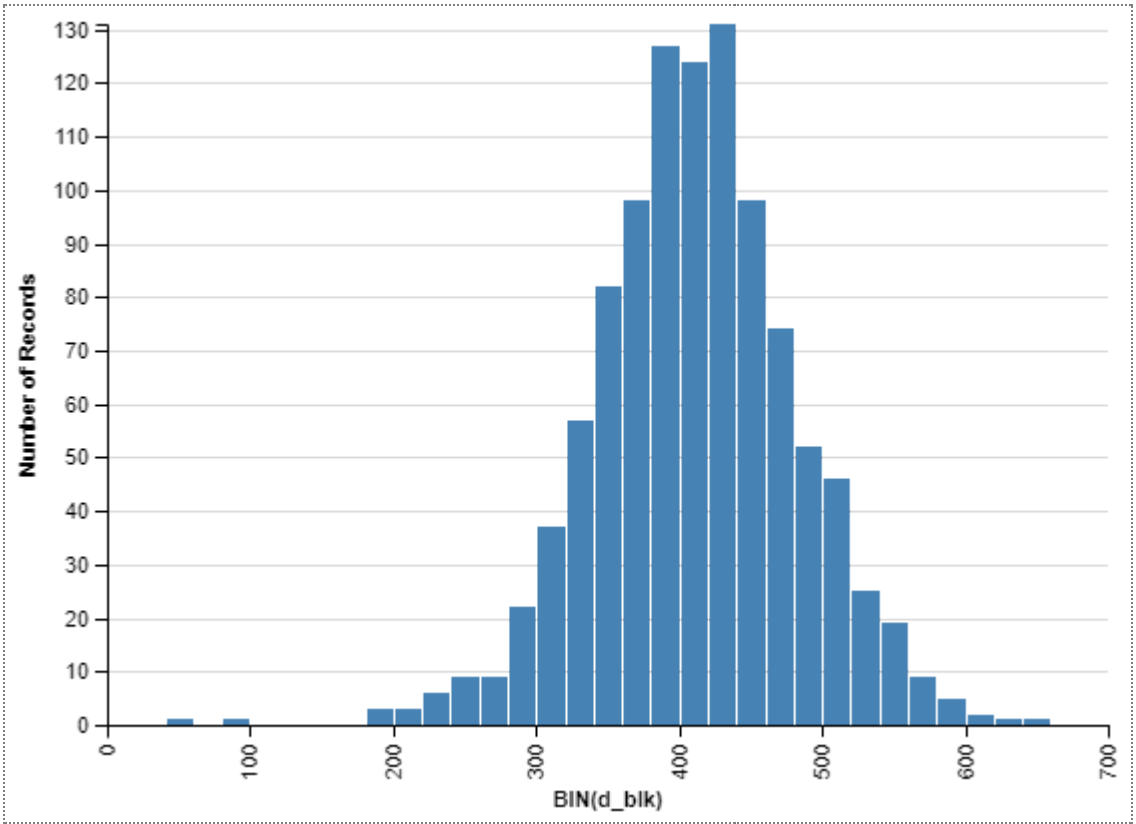


Export as PNG View Source Open in Vega Editor

In [963]:
```
# checking for the missing data in d_blk column in teams_df:
len(teams_df[(teams_df.d_blk.isnull() == True) | (teams_df.d_blk == 0)])
```

Out[963]: 494

In [964]:
```
# Checking for the distribution of d_blk column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_blk.notnull()) & (teams_df.d_blk !=
alt.Chart(teams_df[(teams_df.d_blk.notnull()) & (teams_df.d_blk != 0)]).mark_bar(
    x=alt.X('d_blk', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

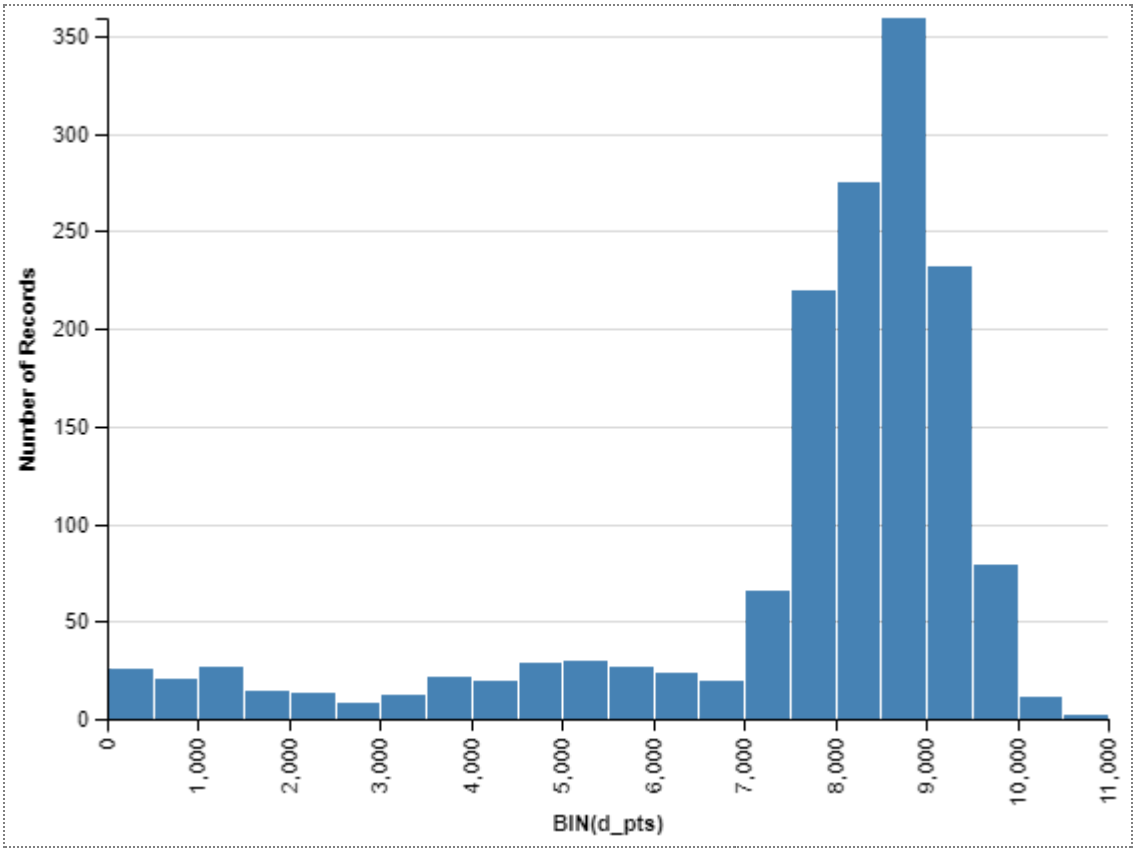


Export as PNG View Source Open in Vega Editor

In [965]:
```
# checking for the missing data in d_pts column in teams_df:
len(teams_df[(teams_df.d_pts.isnull() == True) | (teams_df.d_pts == 0)])
```

Out[965]: 1

In [966]:
```
# Checking for the distribution of d_pts column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.d_pts.notnull()) & (teams_df.d_pts !=
alt.Chart(teams_df[(teams_df.d_pts.notnull()) & (teams_df.d_pts != 0)]).mark_bar(
    x=alt.X('d_pts', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

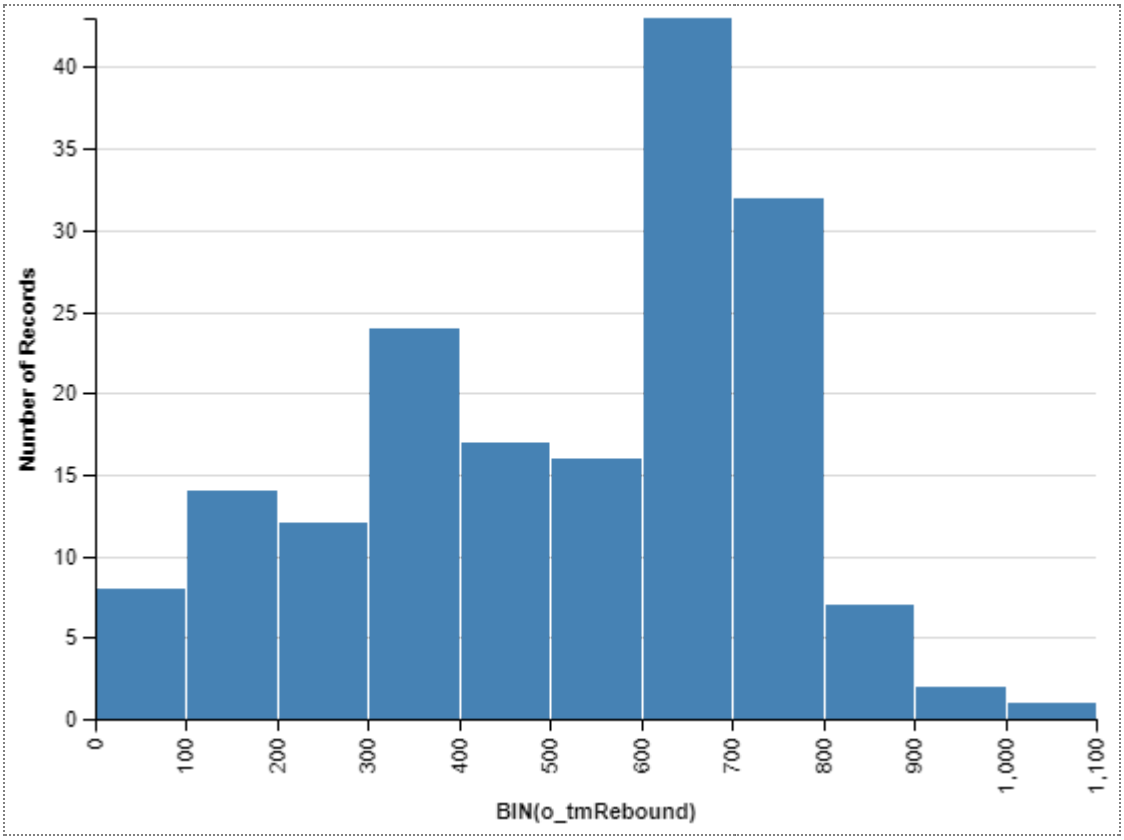


Export as PNG View Source Open in Vega Editor

In [967]:
```
# checking for the missing data in o_tmRebound column in teams_df:
len(teams_df[(teams_df.o_tmRebound.isnull() == True) | (teams_df.o_tmRebound == 0
```

Out[967]: 1360

In [968]:
```
# Checking for the distribution of o_tmRebound column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.o_tmRebound.notnull()) & (teams_df.o_
alt.Chart(teams_df[(teams_df.o_tmRebound.notnull()) & (teams_df.o_tmRebound != 0)
    x=alt.X('o_tmRebound', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```



Export as PNG View Source Open in Vega Editor

In [969]:
```
# checking for the missing data in d_tmRebound column in teams_df:
len(teams_df[(teams_df.d_tmRebound.isnull() == True) | (teams_df.d_tmRebound == 0
```

Out[969]: 1521

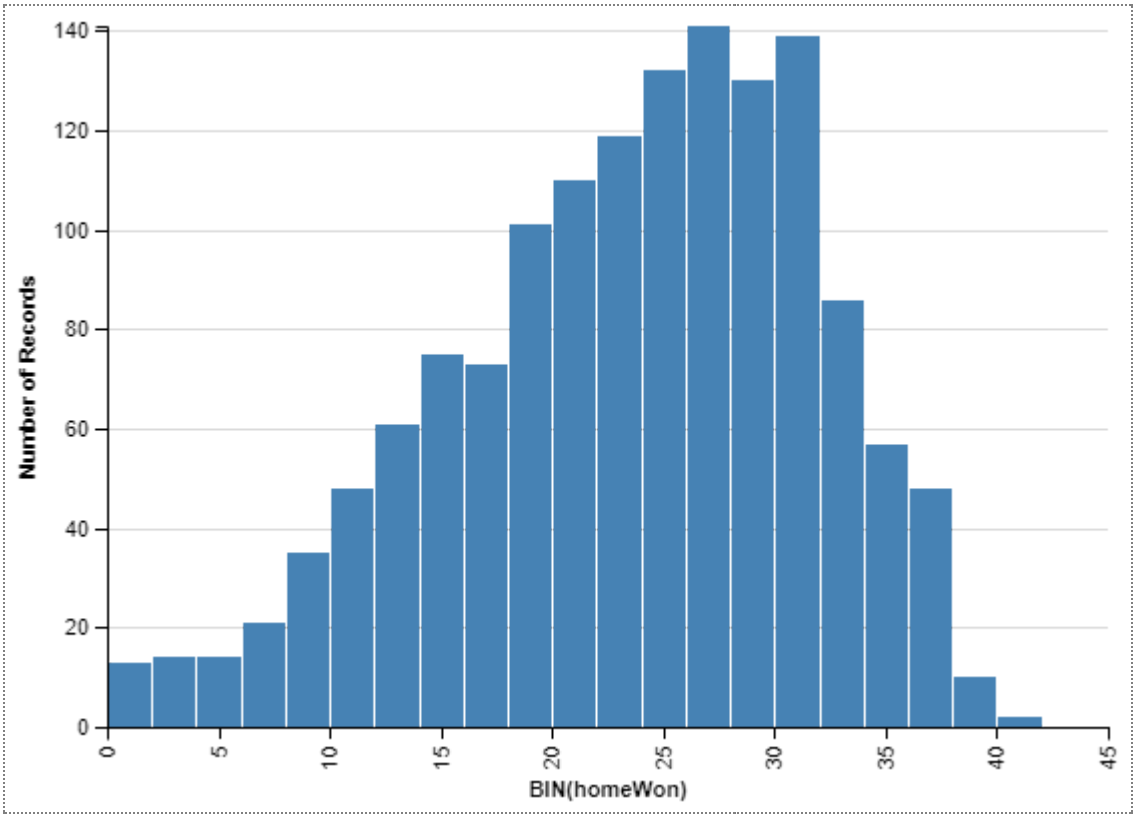
In [970]:
```
# Checking for the distribution of d_tmRebound column in teams_df:
print('The maximum value for the d_tmRebound column is {} and the minimum is {}.'
                        teams_df.d_tmRebound.max(),
    teams_df.d_tmRebound[(teams_df.d_tmRebound.notnull()) & (teams_df.d_tmRebound
```

```
The maximum value for the d_tmRebound column is 769 and the minimum is 385.
```

In [971]:
```
# checking for the missing data in homeWon column in teams_df:
len(teams_df[(teams_df.homeWon.isnull() == True) | (teams_df.homeWon == 0)])
```

Out[971]: 107

In [972]:
```
# Checking for the distribution of homeWon column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.homeWon.notnull()) & (teams_df.homeWo
alt.Chart(teams_df[(teams_df.homeWon.notnull()) & (teams_df.homeWon != 0)]).mark_
    x=alt.X('homeWon', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

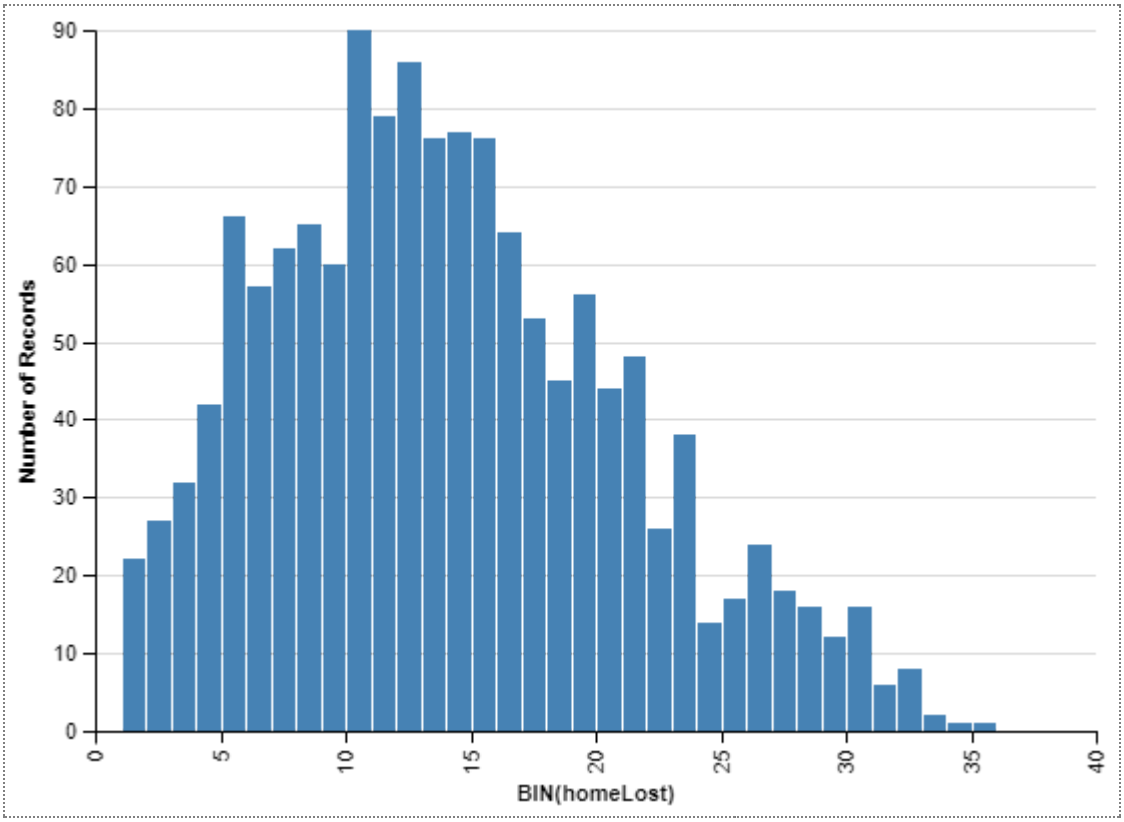


Export as PNG View Source Open in Vega Editor

In [973]:
```
# checking for the missing data in homeLost column in teams_df:
len(teams_df[(teams_df.homeLost.isnull() == True) | (teams_df.homeLost == 0)])
```

Out[973]: 110

In [974]:
```
# Checking for the distribution of homeLost column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.homeLost.notnull()) & (teams_df.homeL
alt.Chart(teams_df[(teams_df.homeLost.notnull()) & (teams_df.homeLost != 0)]).mar
    x=alt.X('homeLost', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

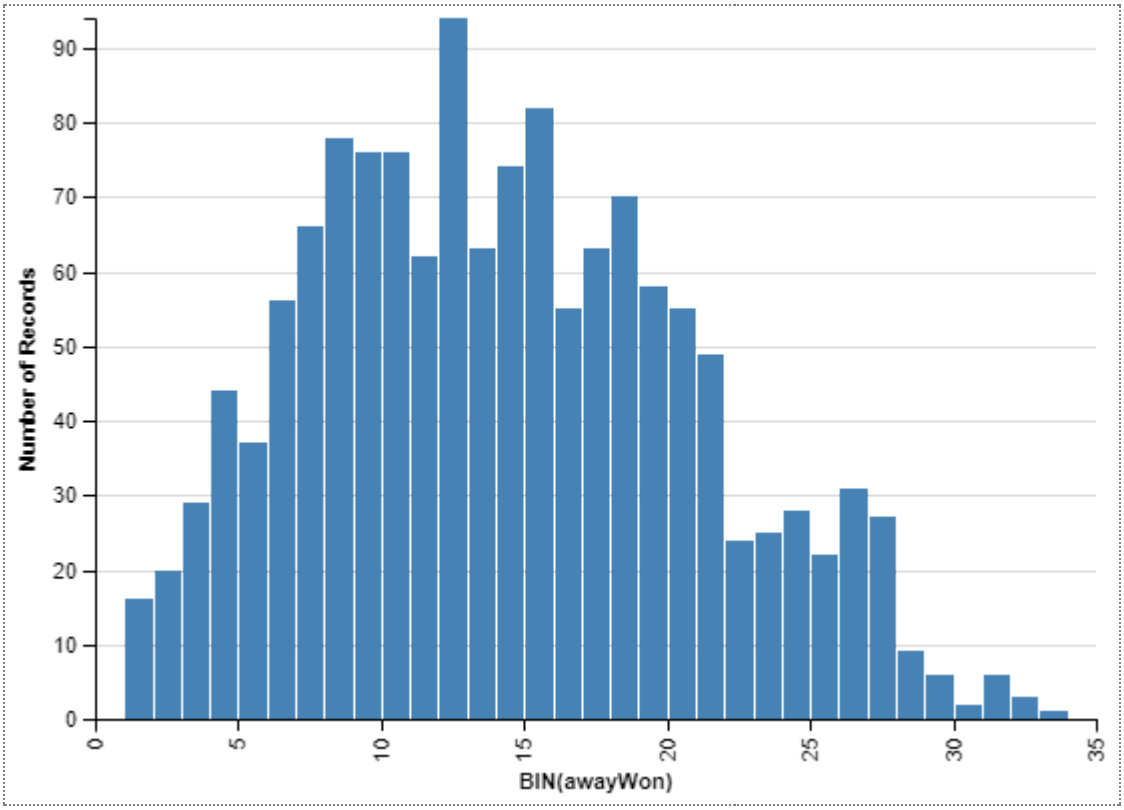


Export as PNG View Source Open in Vega Editor

In [975]:
```
# checking for the missing data in awayWon column in teams_df:
len(teams_df[(teams_df.awayWon.isnull() == True) | (teams_df.awayWon == 0)])
```

Out[975]: 129

In [976]:
```
# Checking for the distribution of awayWon column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.awayWon.notnull()) & (teams_df.awayWo
alt.Chart(teams_df[(teams_df.awayWon.notnull()) & (teams_df.awayWon != 0)]).mark_
    x=alt.X('awayWon', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

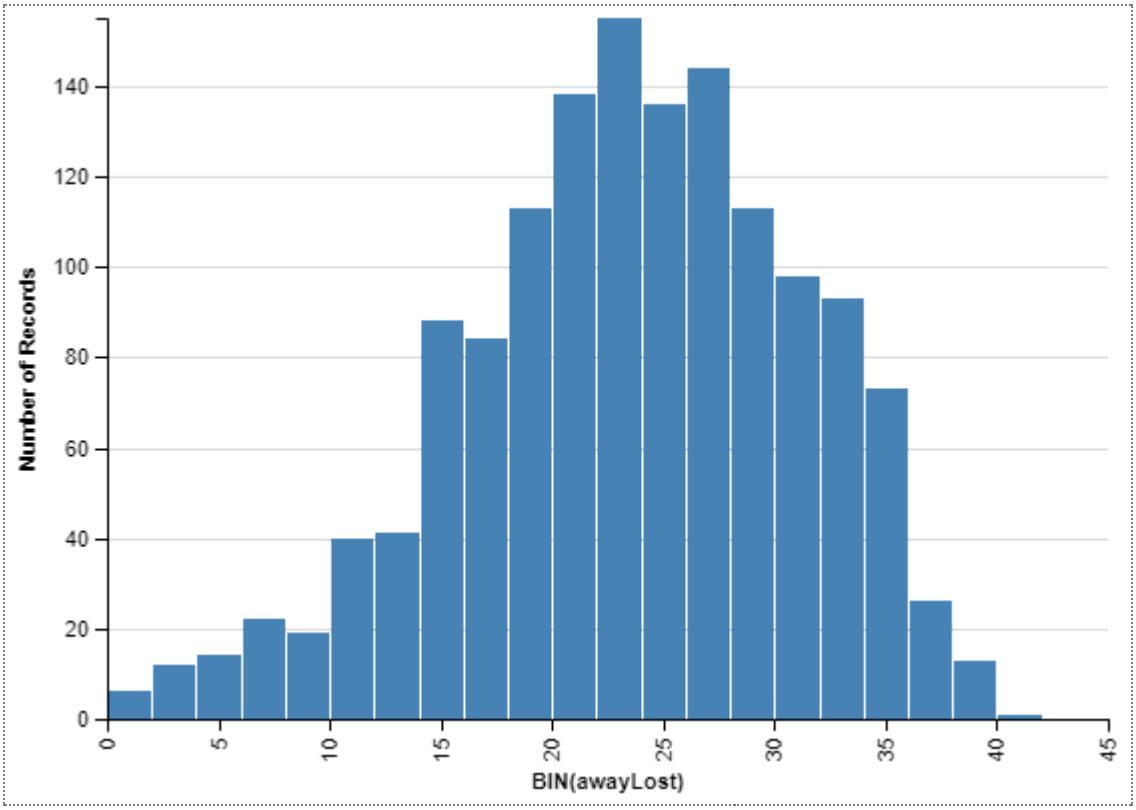


Export as PNG View Source Open in Vega Editor

In [977]:
```
# checking for the missing data in awayLost column in teams_df:
len(teams_df[(teams_df.awayLost.isnull() == True) | (teams_df.awayLost == 0)])
```

Out[977]: 107

In [978]:
```
# Checking for the distribution of awayLost column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.awayLost.notnull()) & (teams_df.awayL
alt.Chart(teams_df[(teams_df.awayLost.notnull()) & (teams_df.awayLost != 0)]).mar
    x=alt.X('awayLost', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```



Export as PNG View Source Open in Vega Editor

In [979]:
```
# checking for the missing data in neutWon column in teams_df:
teams_df.neutWon.isnull().sum()
```

Out[979]: 0

In [980]:
```
# Checking for the distribution of neutWon column in teams_df:
print('The maximum value for the neutWon column is {} and the minimum is {}.'.for
                          teams_df.neutWon.max(),
    teams_df.neutWon.min()))
```

```
The maximum value for the neutWon column is 18 and the minimum is 0.
```

In [981]:
```
# checking for the missing data in neutLoss column in teams_df:
teams_df.neutLoss.isnull().sum()
```

Out[981]: 0

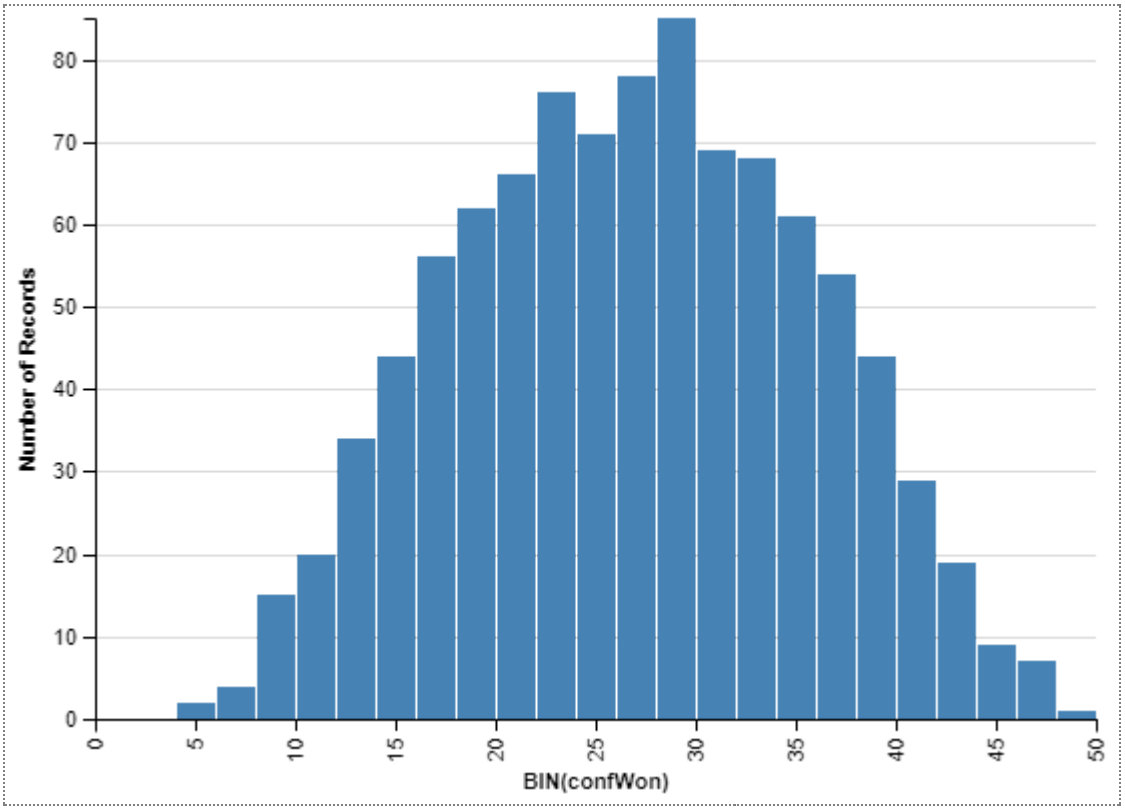
In [982]:
```
# Checking for the distribution of neutLoss column in teams_df:
print('The maximum value for the neutLoss column is {} and the minimum is {}.'.fo
                          teams_df.neutLoss.max(),
    teams_df.neutLoss.min()))
```

```
The maximum value for the neutLoss column is 23 and the minimum is 0.
```

In [983]:
```
# checking for the missing data in confWon column in teams_df:
len(teams_df[(teams_df.awayLost.isnull() == True) | (teams_df.awayLost == 0)])
```

Out[983]:
```
107
```

In [984]:
```
# Checking for the distribution of confWon column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.confWon.notnull()) & (teams_df.confWo
alt.Chart(teams_df[(teams_df.confWon.notnull()) & (teams_df.confWon != 0)]).mark_
    x=alt.X('confWon', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

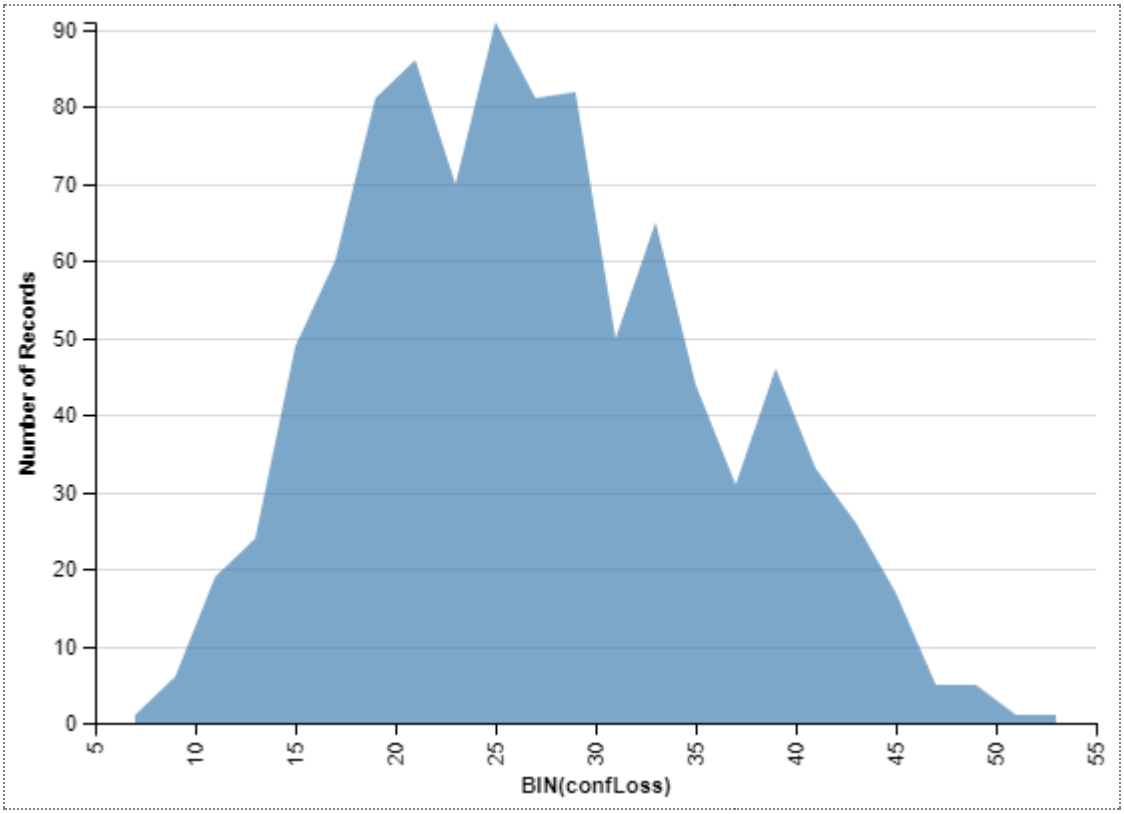


Export as PNG View Source Open in Vega Editor

In [985]:
```
# checking for the missing data in confLoss column in teams_df:
teams_df.confLoss.isnull().sum()
```

Out[985]:
```
0
```

In [986]:
```
# Checking for the distribution of confLoss column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.confLoss.notnull()) & (teams_df.confL
alt.Chart(teams_df[(teams_df.confLoss.notnull()) & (teams_df.confLoss != 0)]).mar
    x=alt.X('confLoss', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

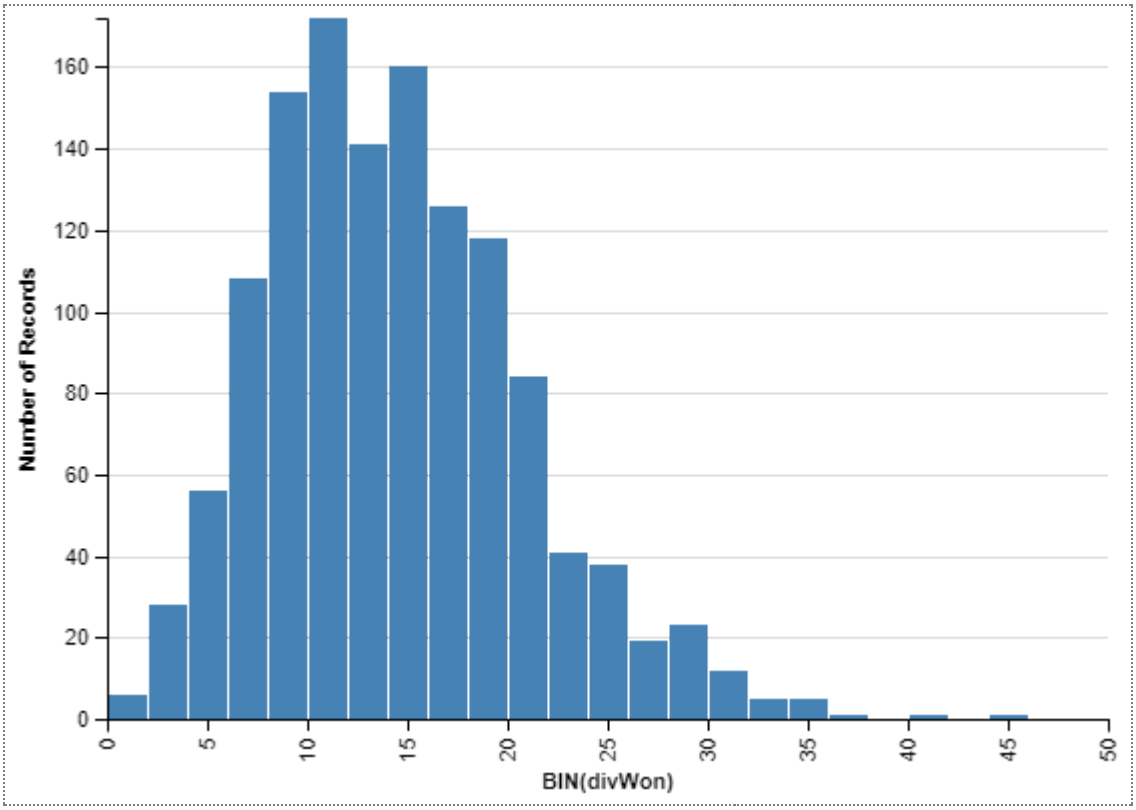


Export as PNG View Source Open in Vega Editor

In [987]:
```
# checking for the missing data in divWon column in teams_df:
len(teams_df[(teams_df.divWon.isnull() == True) | (teams_df.divWon == 0)])
```

Out[987]: 237

In [988]:
```
# Checking for the distribution of divWon column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.divWon.notnull()) & (teams_df.divWon
alt.Chart(teams_df[(teams_df.divWon.notnull()) & (teams_df.divWon != 0)]).mark_ba
    x=alt.X('divWon', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

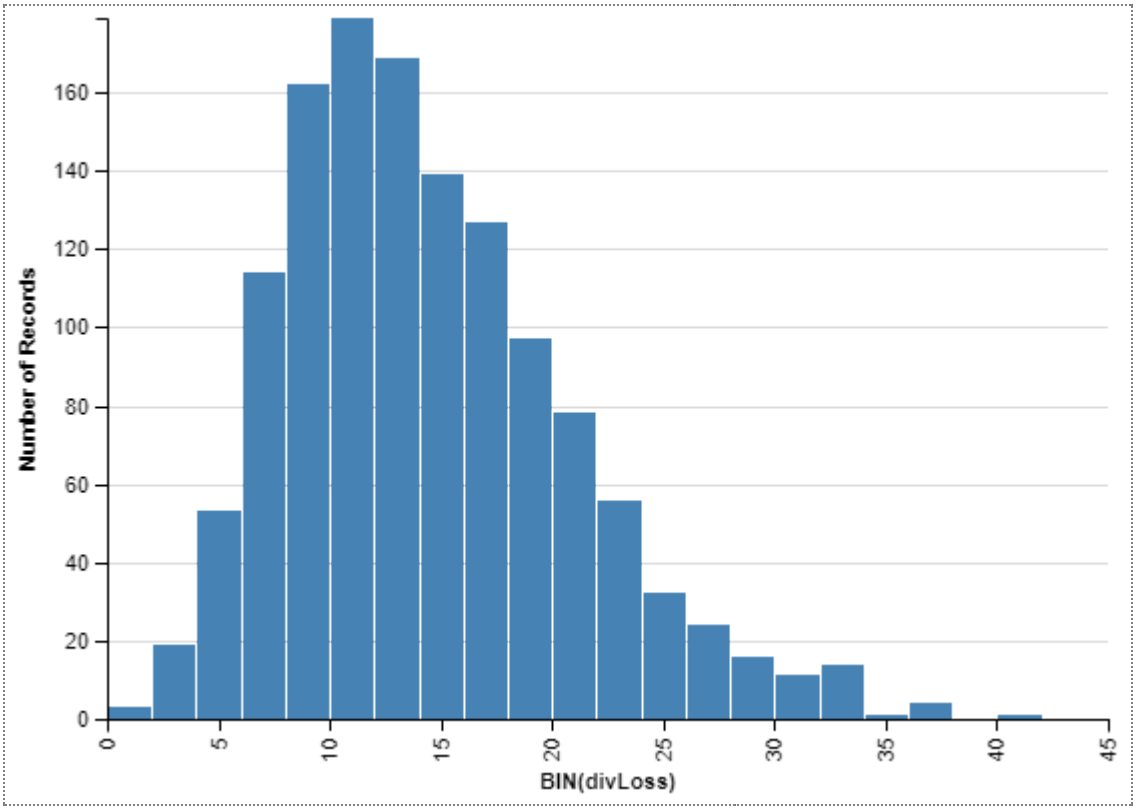


Export as PNG View Source Open in Vega Editor

In [989]:
```
# checking for the missing data in divLoss column in teams_df:
len(teams_df[(teams_df.divLoss.isnull() == True) | (teams_df.divLoss == 0)])
```

Out[989]: 237

In [990]:
```
# Checking for the distribution of divLoss column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.divLoss.notnull()) & (teams_df.divLos
alt.Chart(teams_df[(teams_df.divLoss.notnull()) & (teams_df.divLoss != 0)]).mark_
    x=alt.X('divLoss', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

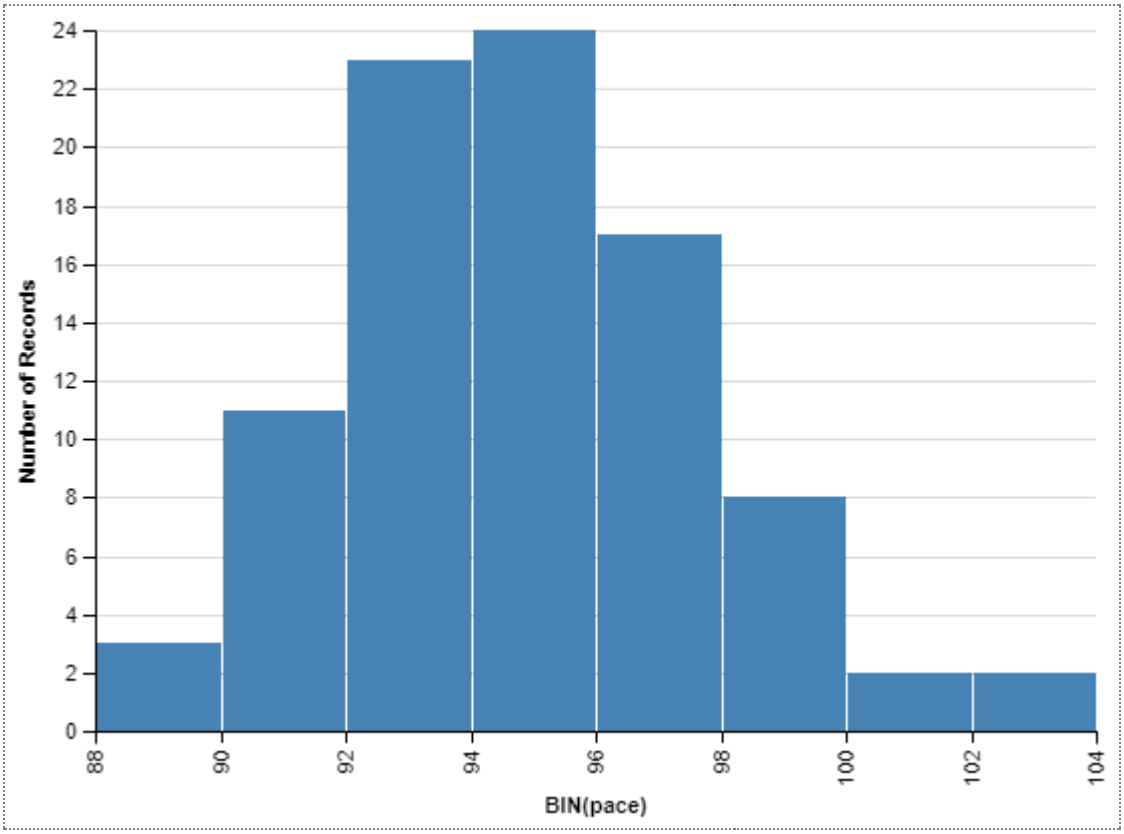


Export as PNG View Source Open in Vega Editor

In [991]:
```
# checking for the missing data in pace column in teams_df:
len(teams_df[(teams_df.pace.isnull() == True) | (teams_df.pace == 0)])
```

Out[991]: 1446

In [992]:
```
# Checking for the distribution of pace column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.pace.notnull()) & (teams_df.pace != 0
alt.Chart(teams_df[(teams_df.pace.notnull()) & (teams_df.pace != 0)]).mark_bar().
    x=alt.X('pace', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```

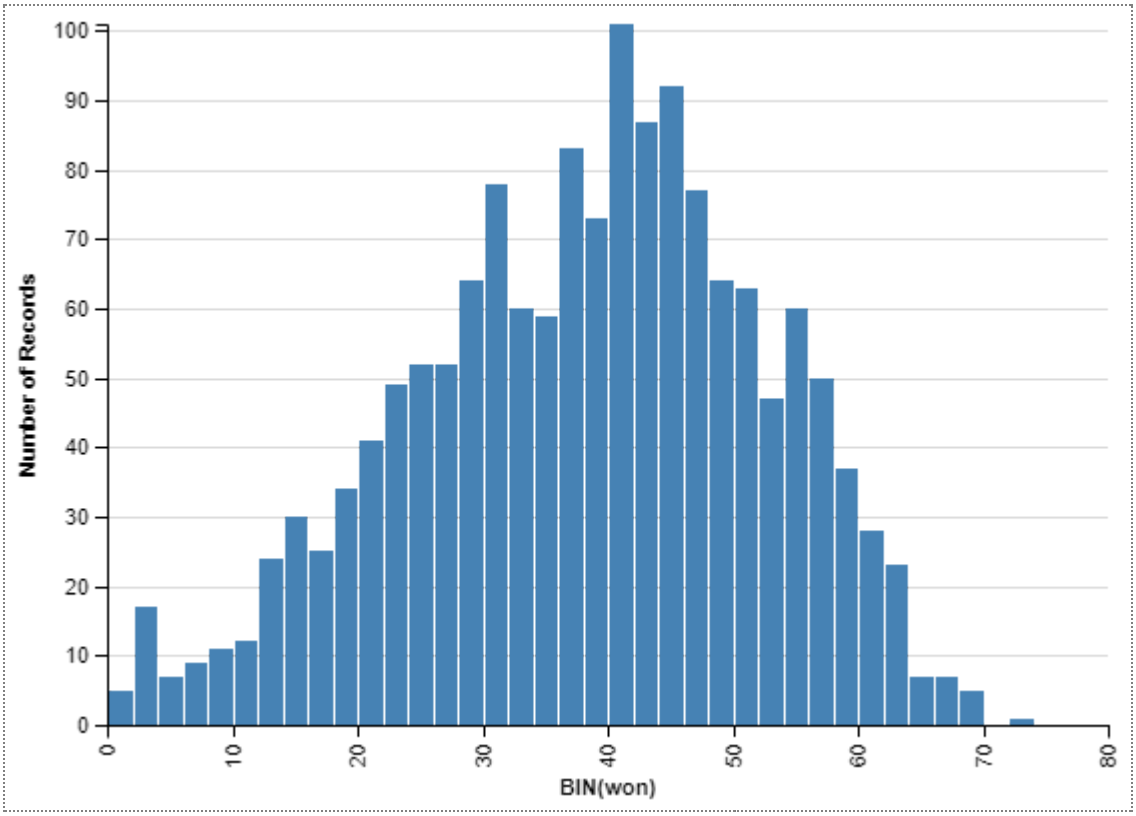


Export as PNG View Source Open in Vega Editor

In [993]:
```
# checking for the missing data in won column in teams_df:
len(teams_df[(teams_df.won.isnull() == True) | (teams_df.won == 0)])
```

Out[993]: 2

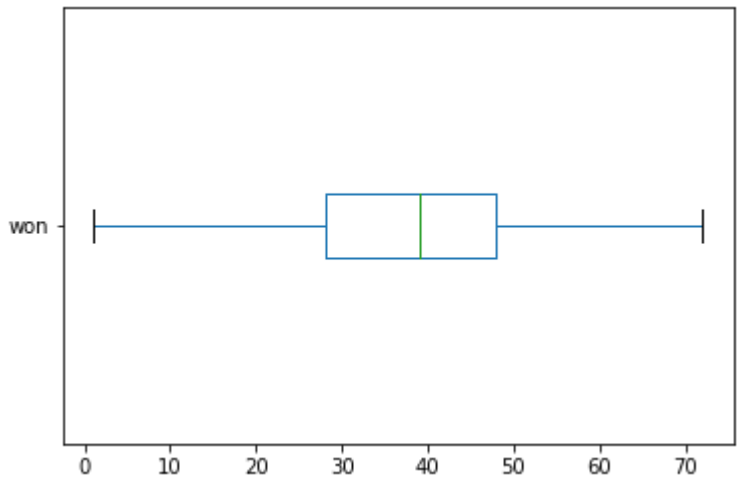
In [994]:
```
# Checking for the distribution of won column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.won.notnull()) & (teams_df.won != 0)]
alt.Chart(teams_df[(teams_df.won.notnull()) & (teams_df.won != 0)]).mark_bar().en
    x=alt.X('won', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```



Export as PNG View Source Open in Vega Editor

In [995]:
```
teams_df.won[(teams_df.won.notnull()) & (teams_df.won != 0)].plot.box(vert=False)
```

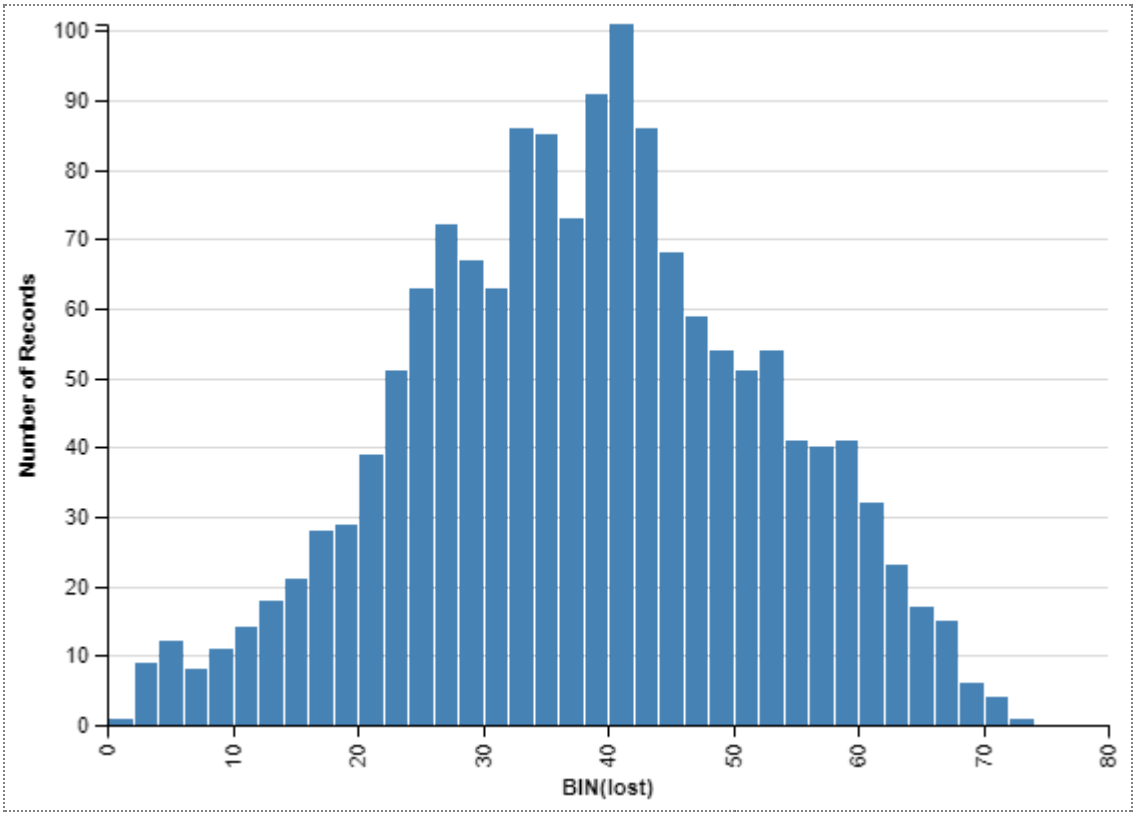
Out[995]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x14b3153ebe0>
```



In [996]:
```
# checking for the missing data in lost column in teams_df:
len(teams_df[(teams_df.lost.isnull() == True) | (teams_df.lost == 0)])
```

Out[996]:
```
2
```

In [997]:
```
# Checking for the distribution of lost column in teams_df:
n_bins = int(np.sqrt(len(teams_df[(teams_df.lost.notnull()) & (teams_df.lost != 0
alt.Chart(teams_df[(teams_df.lost.notnull()) & (teams_df.lost != 0)]).mark_bar().
    x=alt.X('lost', bin= alt.Bin(maxbins=n_bins)), y='count(*)')
```



Export as PNG View Source Open in Vega Editor

In [998]:
```
# checking for the missing data in games column in teams_df:
print(len(teams_df.games[(teams_df.games.isnull() == True) | (teams_df.games == 0
# we can reduce that missing data number by adding up lost games and won games
teams_df.games = teams_df.won + teams_df.lost
print(len(teams_df.games[(teams_df.games.isnull() == True) | (teams_df.games == 0
```

```
90
0
```

In [999]:
```
# Checking for the distribution of games column in teams_df:
print('The maximum value for the games column is {} and the minimum is {}.'.forma
                        teams_df.games.max(),
    teams_df.games[(teams_df.games.notnull()) & (teams_df.games != 0)].min()))
```

```
The maximum value for the games column is 84 and the minimum is 2.
```

In [1000]:
```
# checking for the missing data in min column in teams_df:
len(teams_df.games[(teams_df['min'].isnull() == True) | (teams_df['min'] == 0)])
```

Out[1000]:
```
214
```

In [1001]:
```
# Checking for the distribution of min column in teams_df:
print('The maximum value for the min column is {} and the minimum is {}.'.format(
                        teams_df['min'].max(),
    teams_df['min'][(teams_df['min'].notnull()) & (teams_df.games != 0)].min()))
```

```
The maximum value for the min column is 20460.0 and the minimum is 2640.0.
```

In [1002]:
```
# checking for the missing data in arena column in teams_df:
teams_df.arena.isnull().sum()
```

Out[1002]:
```
189
```

In [1003]:
```
# Checking for the distribution of arena column in teams_df:
print(teams_df.arena.value_counts().head(),
      '\n',
     'Total number of uniuqe values is {}'.format(teams_df.arena.value_counts().c
```

```
Boston Garden                           49
Madison Square Garden (IV)              41
Oakland-Alameda County Coliseum Arena   32
Chicago Stadium                         31
The Spectrum                            27
Name: arena, dtype: int64
 Total number of uniuqe values is 165
```

In [1004]:
```
# checking for the missing data in draftYear column in drafts_df:
len(drafts_df.draftYear[(drafts_df.draftYear.isnull() == True) | (drafts_df.draft
```

Out[1004]:
```
0
```

In [1005]:
```
# Checking for the distribution of draftYear column in drafts_df:
print('The maximum value for the draftYear column is {} and the minimum is {}.'.f
                        drafts_df.draftYear.max(), drafts_df.draftYear.min()))
```

```
The maximum value for the draftYear column is 2011 and the minimum is 1947.
```

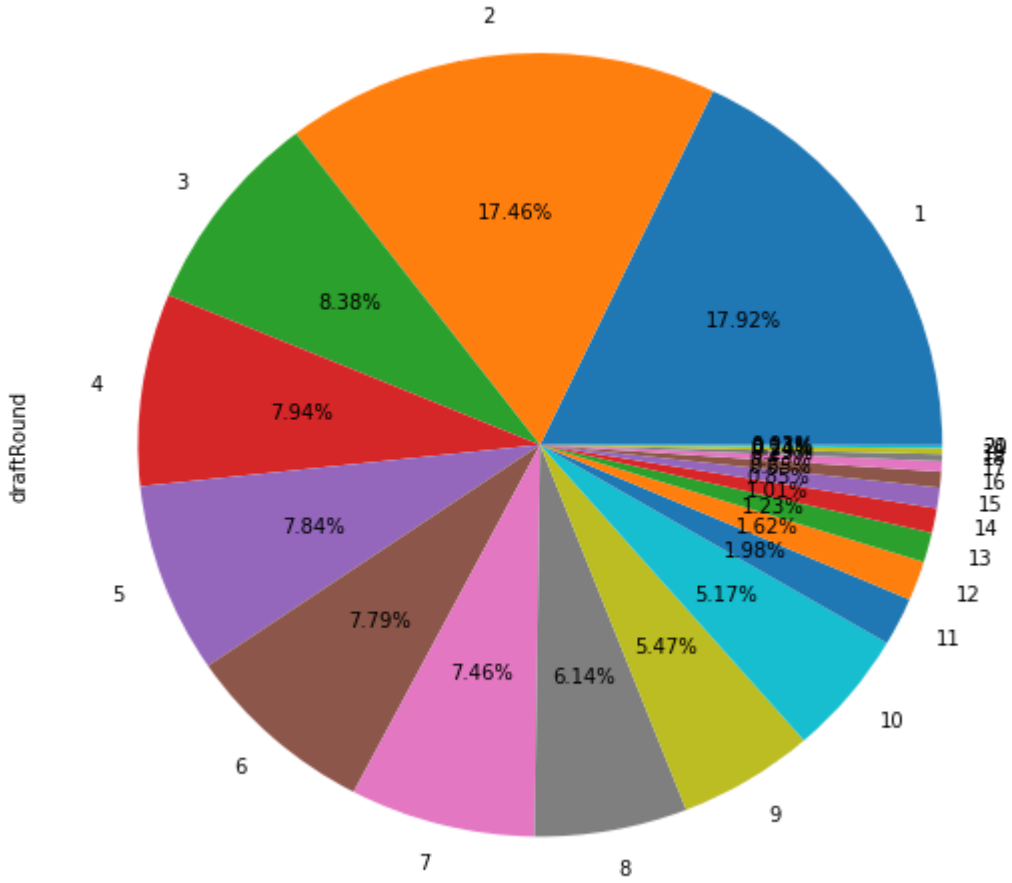
In [1006]:
```
# checking for the missing data in draftRound column in drafts_df:
len(drafts_df[(drafts_df.draftRound.isnull() == True) | (drafts_df.draftRound ==
```

Out[1006]:
```
1493
```

In [1007]:
```
# Checking for the distribution of draftRound column in drafts_df:
print(drafts_df.draftRound.unique())
drafts_df.draftRound[
    (drafts_df.draftRound.isnull() == False)
    & (drafts_df.draftRound != 0)
    & (drafts_df.draftRound != 99)
                            ].value_counts().plot(kind='pie', autopct='%1.2f%%',f
# excluding the 0 and 99 values since they count as a missing data.
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 99]
```

Out[1007]: <matplotlib.axes._subplots.AxesSubplot at 0x14b316c4eb8>



In [1008]:
```
# checking for the missing data in draftSelection column in drafts_df:
len(drafts_df[(drafts_df.draftSelection.isnull() == True) | (drafts_df.draftSelec
```

Out[1008]: 2167

In [1009]:
```
# Checking for the distribution of draftSelection column in drafts_df:
print(drafts_df.draftSelection.value_counts().head(),
      '\n',
     'Total number of uniuqe values is {}'.format(drafts_df.draftSelection.value_
```

```
0    2167
1     514
2     489
3     464
4     445
Name: draftSelection, dtype: int64
 Total number of uniuqe values is 31
```

In [1010]:
```
# checking for the missing data in draftOverall column in drafts_df:
len(drafts_df[(drafts_df.draftOverall.isnull() == True) | (drafts_df.draftOverall
```

Out[1010]: 2167

In [1011]:
```
# Checking for the distribution of draftOverall column in drafts_df:
print('The maximum value for the draftOverall column is {} and the minimum is {}.
                         drafts_df.draftOverall.max(), drafts_df.draftOverall[
                             (drafts_df.draftOverall.isnull() == True) |
                             (drafts_df.draftOverall != 0)].min()))
```

```
The maximum value for the draftOverall column is 239 and the minimum is 1.
```

In [1012]:
```
# checking for the missing data in tmID column in drafts_df:
drafts_df.tmID.isnull().sum()
```

Out[1012]: 0

In [1013]:
```
# Checking for the distribution of tmID column in drafts_df:
print(drafts_df.tmID.value_counts().head(),
      '\n',
     'Total number of uniuqe values is {}'.format(drafts_df.tmID.value_counts().c
```

```
NYK    482
BOS    440
PHI    381
DET    352
CHI    326
Name: tmID, dtype: int64
 Total number of uniuqe values is 96
```

In [1014]:
```
# checking for the missing data in firstName column in drafts_df:
drafts_df.firstName.isnull().sum()
```

Out[1014]: 0

In [1015]:
```
# Checking for the distribution of firstName column in drafts_df:
print(drafts_df.firstName.value_counts().head(),
      '\n',
     'Total number of uniuqe values is {}'.format(drafts_df.firstName.value_count
```

```
John    271
Bob     260
Jim     259
Mike    223
Bill    197
Name: firstName, dtype: int64
 Total number of uniuqe values is 1465
```

In [1016]:
```
# checking for the missing data in lastName column in drafts_df:
drafts_df.lastName.isnull().sum()
```

Out[1016]: 0

In [1017]:
```
# Checking for the distribution of lastName column in drafts_df:
print(drafts_df.lastName.value_counts().head(),
      '\n',
     'Total number of uniuqe values is {}'.format(drafts_df.lastName.value_counts
```

```
Smith       136
Williams    123
Johnson     119
Jones       109
Davis        80
Name: lastName, dtype: int64
 Total number of uniuqe values is 4193
```

In [1018]:
```
# checking for the missing data in playerID column in drafts_df:
drafts_df.playerID.isnull().sum()
```

Out[1018]: 5189

In [1019]:
```
# Checking for the distribution of playerID column in drafts_df:
print(drafts_df.playerID.value_counts().head(),
      '\n',
     'Total number of uniuqe values is {}'.format(drafts_df.playerID.value_counts
```

```
catchha01    5
jonesdw01    4
cheniph01    4
thomaro01    4
edgech01     4
Name: playerID, dtype: int64
 Total number of uniuqe values is 3133
```

In [1020]:
```
# checking for the missing data in draftFrom column in drafts_df:
drafts_df.draftFrom.isnull().sum()
```

Out[1020]: 6

In [1021]:
```
# Checking for the distribution of draftFrom column in drafts_df:
print(drafts_df.draftFrom.value_counts().head(),
      '\n',
     'Total number of uniuqe values is {}'.format(drafts_df.draftFrom.value_count
```

```
UCLA             125
Kentucky         123
North Carolina   113
Duke              90
Kansas            85
Name: draftFrom, dtype: int64
 Total number of uniuqe values is 1266
```

In [1022]:
```
# checking for the missing data in lgID column in drafts_df:
drafts_df.lgID.isnull().sum()
```

Out[1022]:
```
0
```

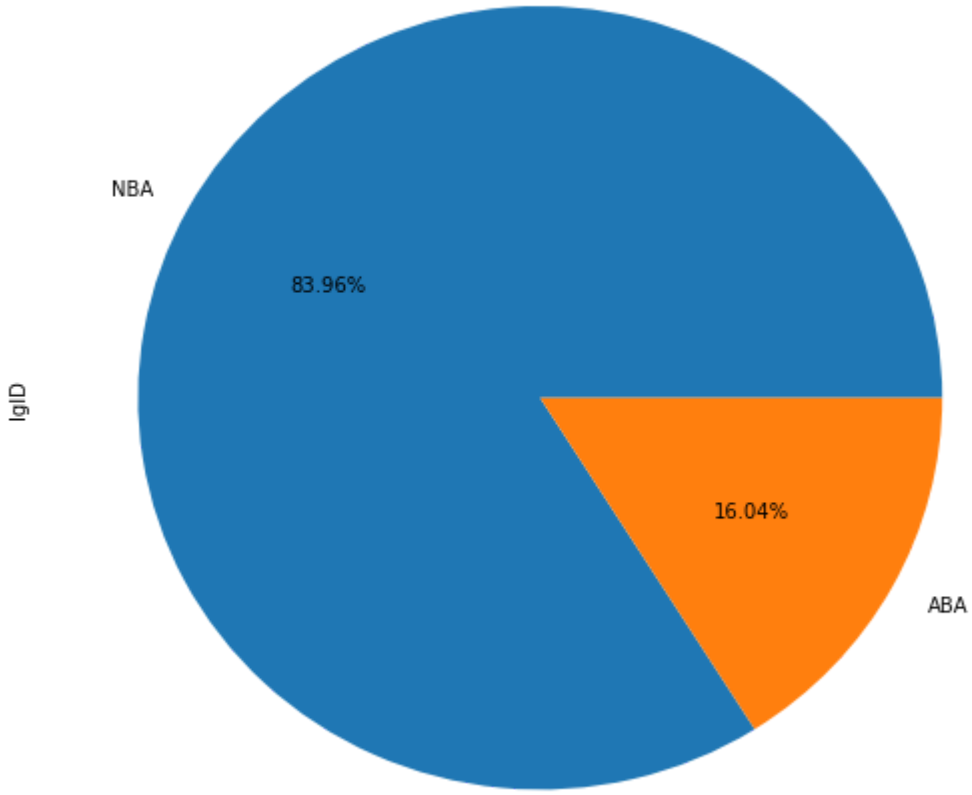
In [1023]:
```
# Checking for the distribution of lgID column in drafts_df:
print(drafts_df.lgID.value_counts().head(),
      '\n',
     'Total number of uniuqe values is {}'.format(drafts_df.lgID.value_counts().c
```

```
NBA    7559
ABA    1444
Name: lgID, dtype: int64
 Total number of uniuqe values is 2
```

In [1024]:
```
# Checking for the distribution of lgID column in drafts_df:
print(drafts_df.lgID.unique())
drafts_df.lgID[
    (drafts_df.draftRound.isnull() == False)].value_counts().plot(kind='pie', aut
```

```
['ABA' 'NBA']
```

Out[1024]: <matplotlib.axes._subplots.AxesSubplot at 0x14b31db5f60>



In [1025]:
```
# we can create new column in drafts_df that we call draftCountry,
# where we can find the country that the player has been drafted from
# by taking the values within the parenthesis.
```

In [1026]:
```
# this is a list of all countries
req = requests.get('https://rawgit.com/ryankane/d40364df707e9e600fcaa0bb645c762f/
countries = (i['name'] for i in req)
countires_lower = [x.lower() for x in list(countries)]

def slice_country(row):
    if row:
        if re.search('\((.*?)\)',str(row)) and re.search('\((.*?)\)',str(row)).gr
            return re.search('\((.*?)\)',str(row)).group(1)
        else:
            return 'USA'



drafts_df['draftCountry'] = drafts_df.draftFrom.apply(slice_country)
drafts_df.tail()
```

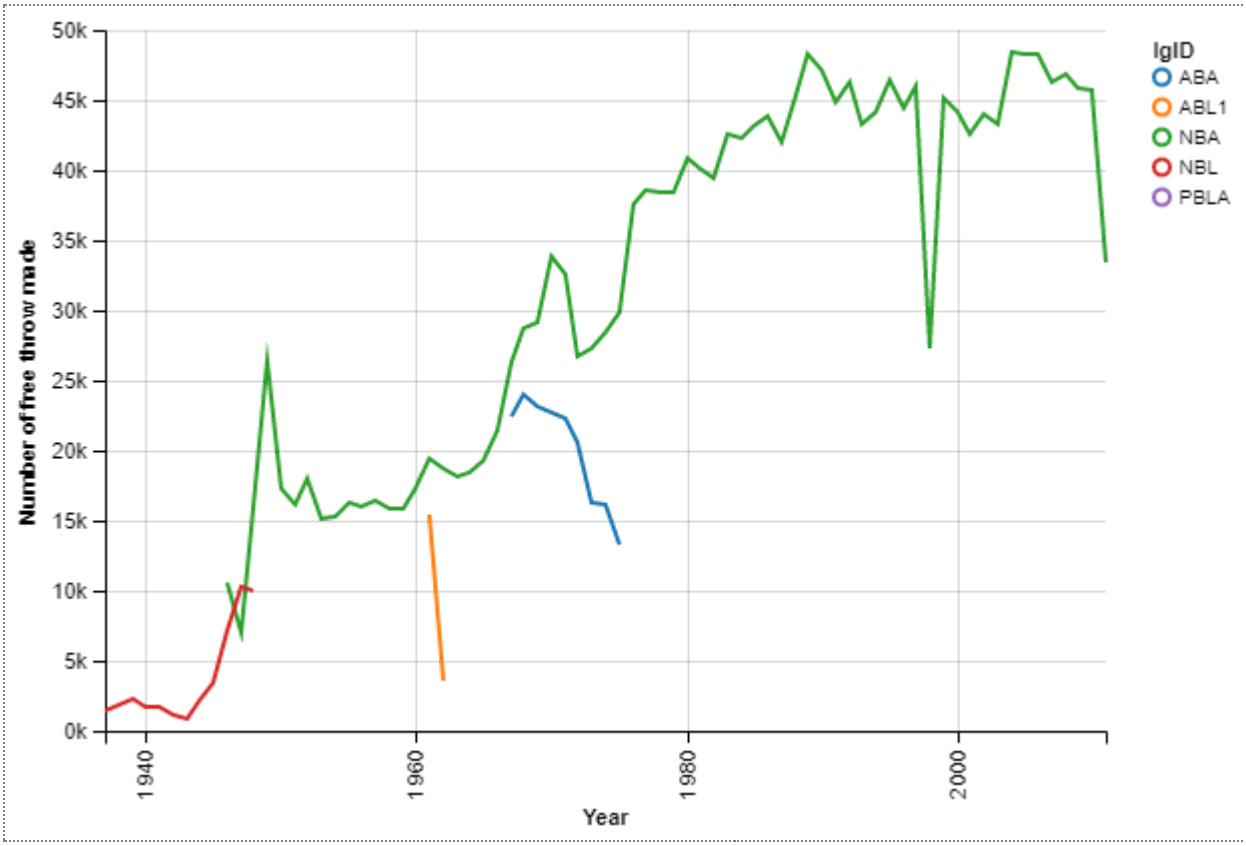
Out[1026]:

	draftYear	draftRound	draftSelection	draftOverall	tmID	firstName	lastName	playerID
8998	2011	2	26	56	LAL	Chukwudiebere	Maduabum	NaN
8999	2011	2	27	57	DAL	Tanguy	Ngombo	NaN
9000	2011	2	28	58	LAL	Ater	Majok	NaN
9001	2011	2	29	59	SAS	Adam	Hanga	NaN
9002	2011	2	30	60	SAC	Isaiah	Thomas	thomais02

In [1027]:
```
len(drafts_df.draftCountry.value_counts())
```

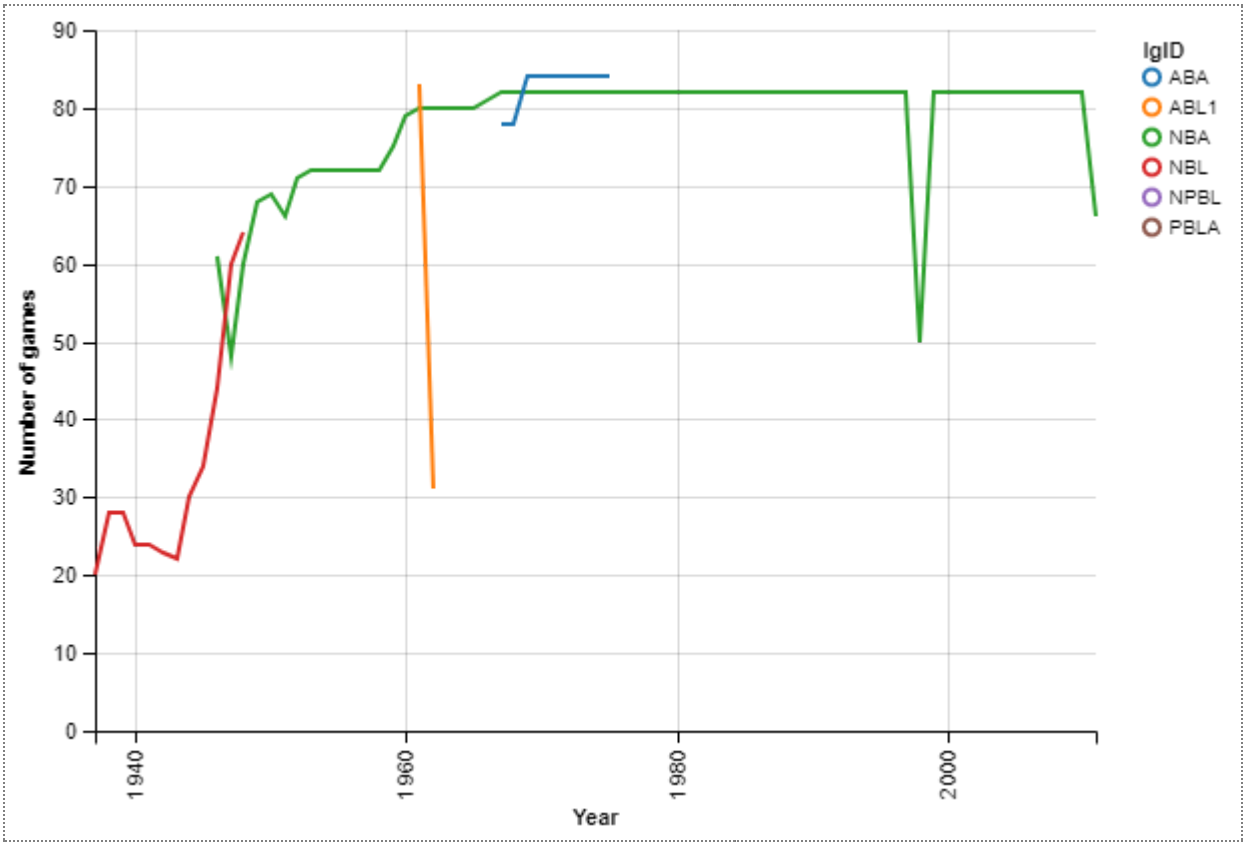
Out[1027]: 30

In [1028]:
```
alt.Chart(teams_df[teams_df.o_ftm != 0]).mark_line().encode(
    x=alt.X('year:T' ,timeUnit='year', title='Year'),y=alt.Y('sum(o_ftm)',title='
```



Export as PNG View Source Open in Vega Editor

In [1029]:
```
alt.Chart(teams_df).mark_line().encode(
    x=alt.X('year:T' ,timeUnit='year',title='Year'),y=alt.Y('max(games)', title='
```



Export as PNG View Source Open in Vega Editor

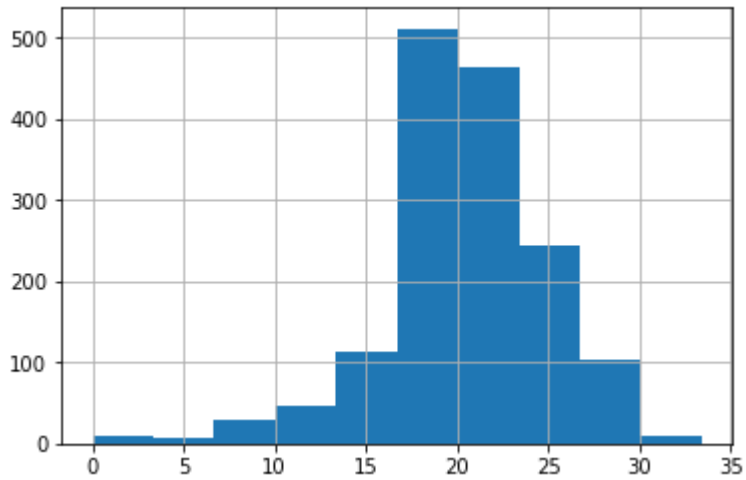
In [1030]:
```
# checking for number of games in the seasons between 1990 and 2000
teams_df.year[(teams_df.games == 50)]
```

Out[1030]:
```
984     1998
985     1998
986     1998
987     1998
988     1998
989     1998
990     1998
991     1998
992     1998
993     1998
994     1998
995     1998
996     1998
997     1998
998     1998
999     1998
1000    1998
1001    1998
1002    1998
1003    1998
1004    1998
1005    1998
1006    1998
1007    1998
1008    1998
1009    1998
1010    1998
1011    1998
1012    1998
Name: year, dtype: int64
```

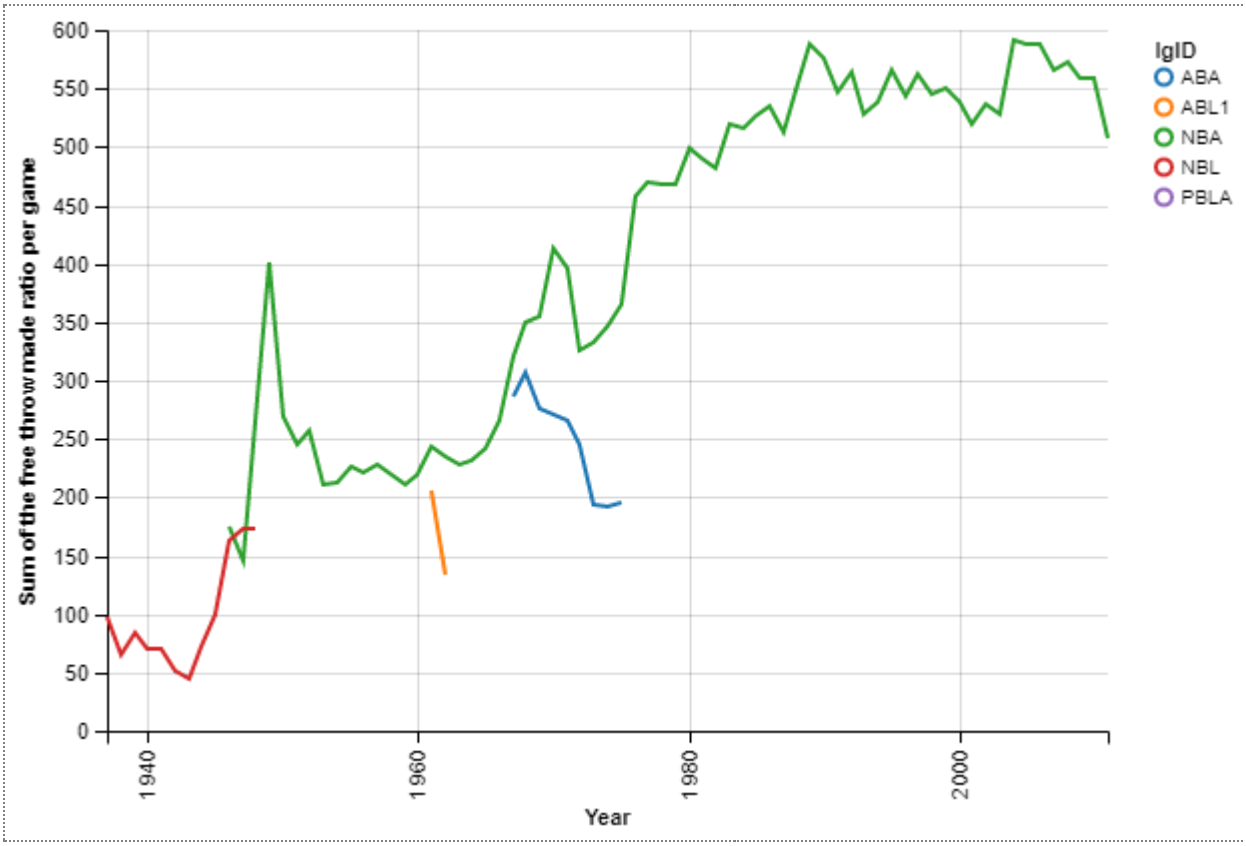
In [1031]:
```
teams_df['o_ftm_games_ratio'] = teams_df.o_ftm / teams_df.games
```

In [1032]:
```
teams_df.o_ftm_games_ratio.hist()
```

Out[1032]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x14b31855f60>
```

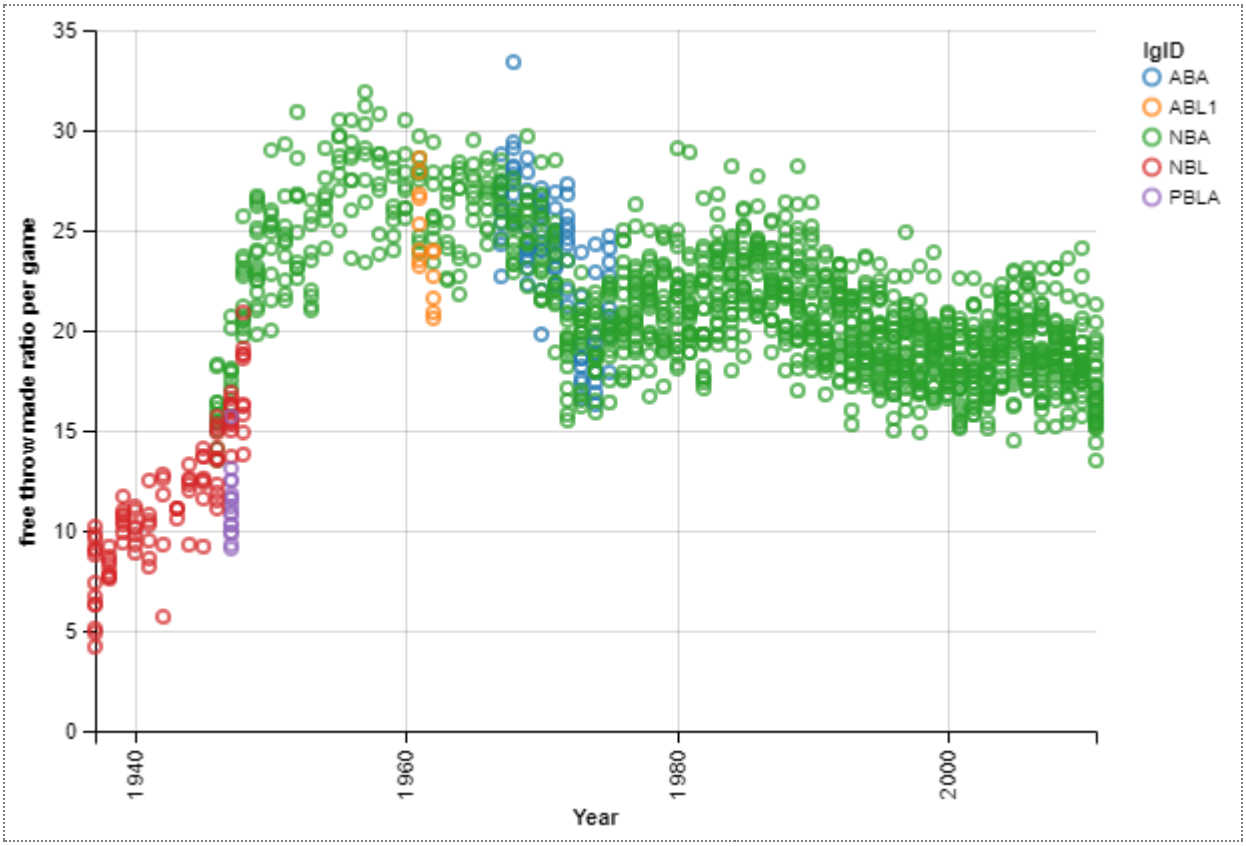


In [1033]:
```
alt.Chart(teams_df[teams_df.o_ftm != 0]).mark_line().encode(
    x=alt.X('year:T' ,timeUnit='year',title='Year'),
    y=alt.Y('sum(o_ftm_games_ratio)',title='Sum of the free throw made ratio per
```



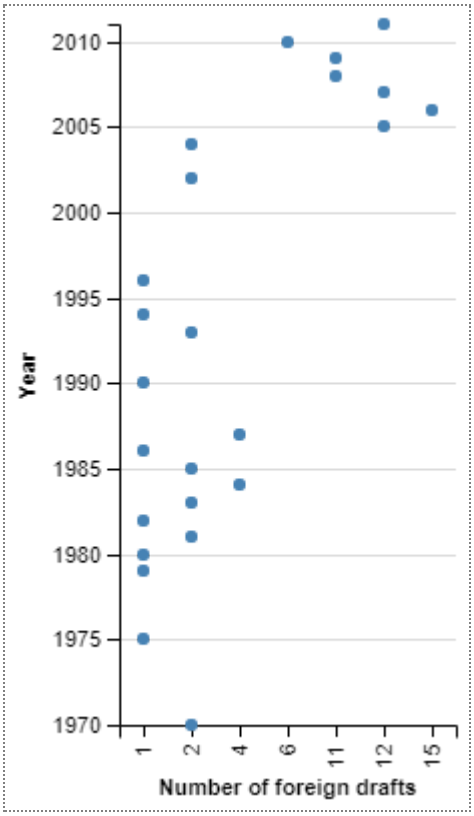
Export as PNG View Source Open in Vega Editor

In [1034]:
```
alt.Chart(teams_df[teams_df.o_ftm != 0]).mark_point().encode(
    x=alt.X('year:T' ,timeUnit='year',title='Year'),
    y=alt.Y('o_ftm_games_ratio',title='free throw made ratio per game'),color='lg
```



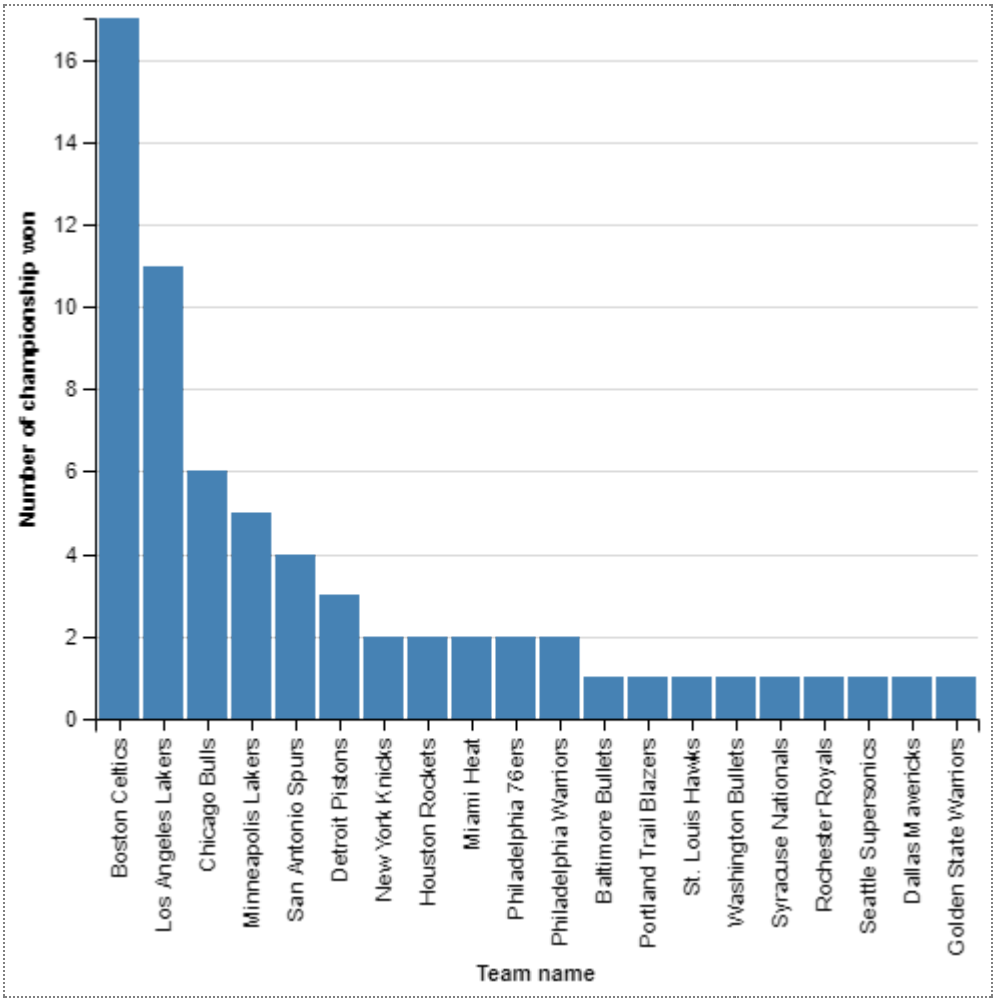
Export as PNG View Source Open in Vega Editor

In [1035]:
```
alt.Chart(drafts_df[drafts_df.draftCountry != 'USA']).mark_circle().encode(
    y=alt.Y('draftYear:T',timeUnit='year',title='Year'),x=alt.X('count(draftCount
```



Export as PNG View Source Open in Vega Editor

In [1036]:
```
# team that has won most nba championship
alt.Chart(teams_df[teams_df.playoff == 'NC']).mark_bar().encode(
    x=alt.X('name', sort= alt.SortField(field='*', op='count', order='descending'
    y=alt.Y('count(*)',title='Number of championship won'))
```



Export as PNG View Source Open in Vega Editor

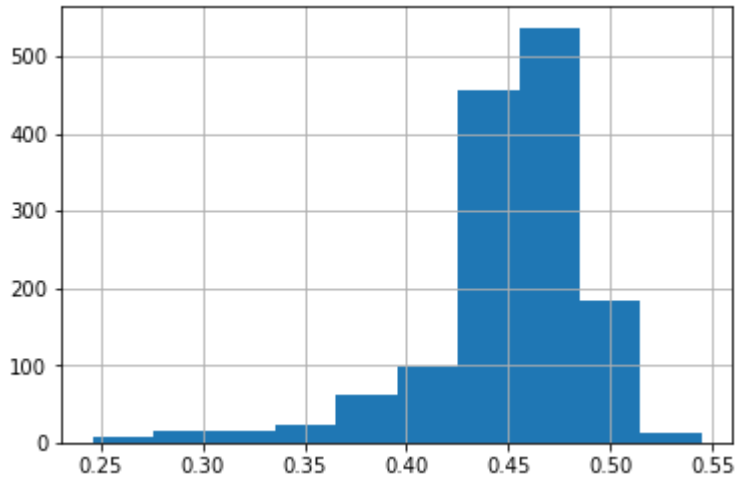
In [1037]:
```
teams_df['o_fgp']=teams_df.o_fgm[(teams_df.o_fgm != 0) & (teams_df.o_fga != 0)] /
teams_df.head()
```

Out[1037]:

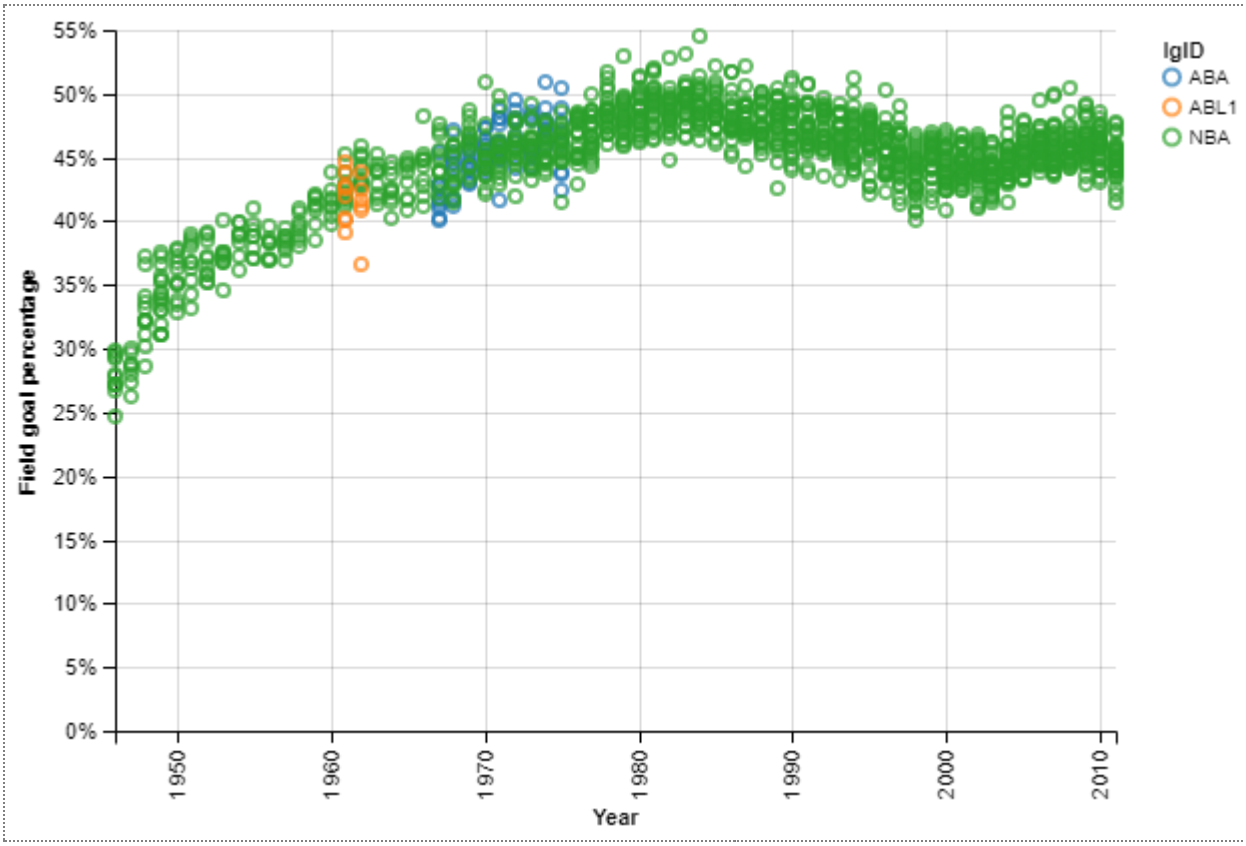
	year	lgID	tmID	franchID	confID	divID	rank	confRank	playoff	name	o_fgm	o_fga	o_
0	1946	NBA	BOS	BOS	NaN	ED	5	0	NaN	Boston Celtics	1397	5133	
1	1946	NBA	CHS	CHS	NaN	WD	1	0	F	Chicago Stags	1879	6309	
2	1946	NBA	CLR	CLR	NaN	WD	3	0	R1	Cleveland Rebels	1674	5699	
3	1946	NBA	DTF	DTF	NaN	WD	4	0	NaN	Detroit Falcons	1437	5843	
4	1946	NBA	NYK	NYK	NaN	ED	3	0	SF	New York Knicks	1465	5255	

In [1038]:
```
teams_df.o_fgp.hist()
```

Out[1038]: <matplotlib.axes._subplots.AxesSubplot at 0x14b33fa58d0>



In [1039]:
```
# checking if the players is being more effection in shooting field goals over th
alt.Chart(teams_df[teams_df.o_fgp != 0]).mark_point().encode(
    x=alt.X('year:T' ,timeUnit='year',title='Year'),
    y=alt.Y('o_fgp:Q',title='Field goal percentage', axis=alt.Axis(format='%')),c
```



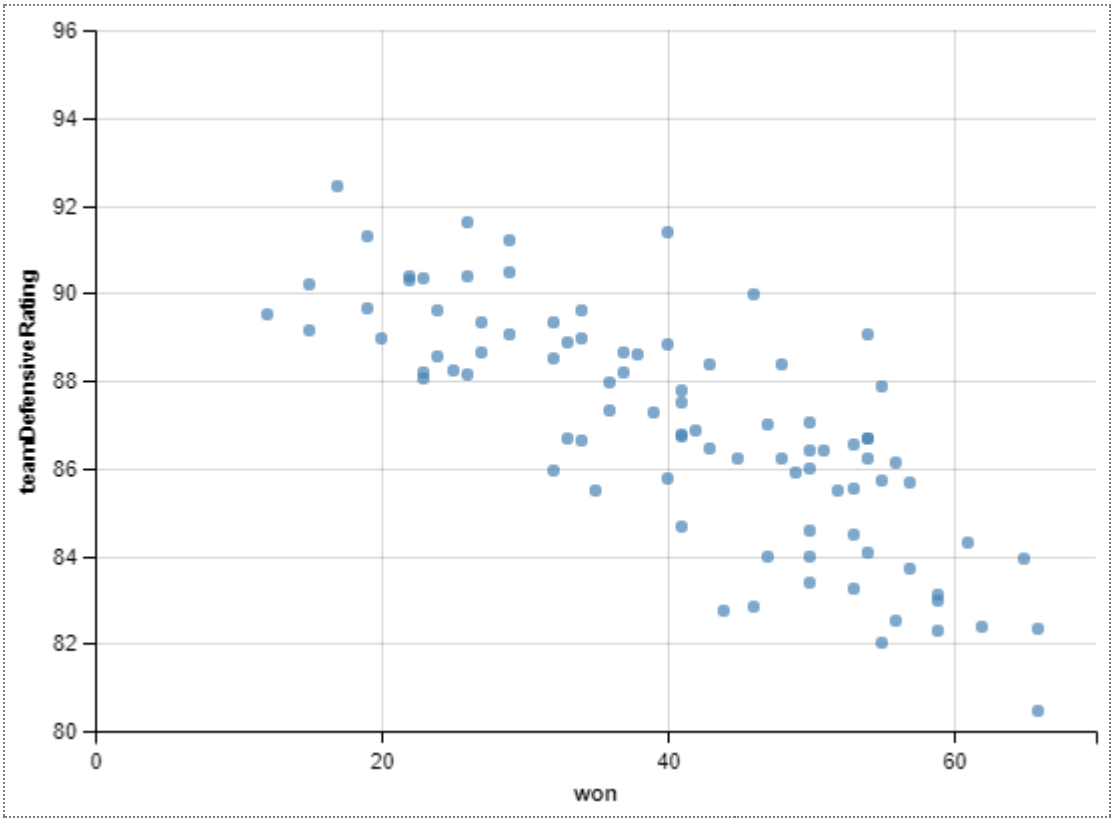
Export as PNG View Source Open in Vega Editor

In [1040]:
```
# Calculating Team_Defensive_Rating ref:https://www.basketball-reference.com/abou
teams_df['teamDefensiveRating']=(teams_df.d_pts[teams_df.pace !=0] / teams_df.pac
teams_df[teams_df.teamDefensiveRating.notnull()].head()
# check the second row. NC = won the nba championship
```

Out[1040]:

	year	lgID	tmID	franchID	confID	divID	rank	confRank	playoff	name	o_fgm	o_fga
1248	2007	NBA	ATL	ATL	EC	SE	3	8	C1	Atlanta Hawks	2975	6552
1249	2007	NBA	BOS	BOS	EC	AT	1	1	NC	Boston Celtics	2986	6286
1250	2007	NBA	CHA	CHA	EC	SE	4	12	NaN	Charlotte Bobcats	2960	6554
1251	2007	NBA	CHI	CHI	EC	CD	4	11	NaN	Chicago Bulls	2987	6861
1252	2007	NBA	CLE	CLE	EC	CD	2	4	CS	Cleveland Cavaliers	2937	6697

In [1041]:
```
# checking the relationship between team defensive rating and how many times did
alt.Chart(teams_df[(teams_df.won.notnull())]).mark_circle().encode(
    x='won',y=alt.Y('teamDefensiveRating',scale=alt.Scale(domain=[80, 95])))
```



Export as PNG View Source Open in Vega Editor

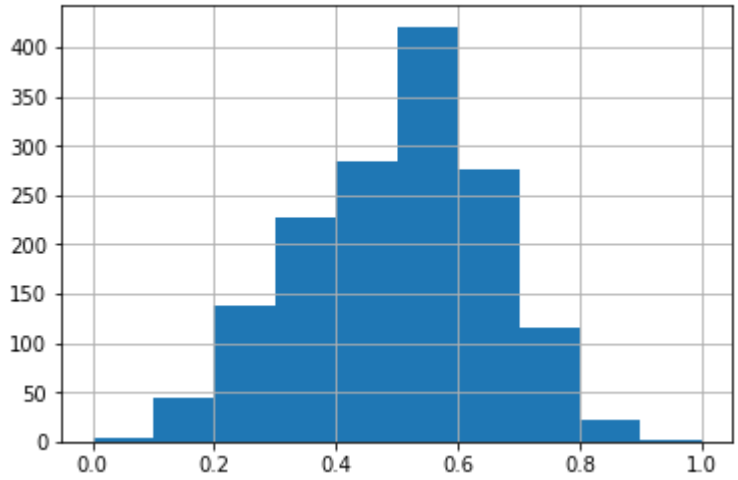
In [1042]:
```
# checking the correlation between team defensive rating and how many times did t
teams_df.teamDefensiveRating.corr(teams_df.won)
```

Out[1042]: -0.78749348034798461

In [1043]:
```
# calculating Won-Lost Percentage
teams_df['wonLostPercentage']=teams_df.won / teams_df.games
```

In [1044]:
```
teams_df.wonLostPercentage.hist()
```

Out[1044]: <matplotlib.axes._subplots.AxesSubplot at 0x14b340aceb8>



In [1045]:
```
# Team who has the highest wonLostPercentage
teams_df[teams_df.wonLostPercentage == teams_df[teams_df.lgID =='NBA'].wonLostPer
```

Out[1045]:

	year	lgID	tmID	franchID	confID	divID	rank	confRank	playoff	name	o_fgm	o_fga	o
900	1995	NBA	CHI	CHI	EC	CD	1	1	NC	Chicago Bulls	3293	6892	

In [1047]:
```
# finding the sum of away and home wins
print(teams_df[(teams_df.awayWon.notnull()) & (teams_df.homeWon.notnull())].awayW
print(teams_df[(teams_df.awayWon.notnull()) & (teams_df.homeWon.notnull())].homeW
```

```
19368
32811
```