In [1]:
```
#import required packages
import pandas as pd
import h2o
from collections import Counter
import numpy as np
from h2o.estimators.gbm import H2OGradientBoostingEstimator
```

In [2]:
```
#Prepare the data frame
data = pd.read_csv('Nashville.csv')
data.columns = ['date', 'name1', 'name2', 'draft', 'color', 'splash', 'la
                'exceed', 'failed', 'camp', 'x1', 'x2']
data['color2'] = [x.replace(' ',  '').split(',') for x in data.color]
data['c1'] = [x.replace(' ',  '').split(',')[0] for x in data.color]
data['c2'] = [x.replace(' ',  '').split(',')[1] if len(x.replace(' ',  ''
data['name'] = data.name1.fillna(data.name2)
data['white'] = [1 if 'White' in colors else 0 for colors in data.color2]
data['blue'] = [1 if 'Blue' in colors else 0 for colors in data.color2]
data['black'] = [1 if 'Black' in colors else 0 for colors in data.color2]
data['red'] = [1 if 'Red' in colors else 0 for colors in data.color2]
data['green'] = [1 if 'Green' in colors else 0 for colors in data.color2]
data['wins'] = [Counter(x)['Win'] for x in zip(data.r1, data.r2, data.r3)
data['losses'] = [Counter(x)['Lose'] for x in zip(data.r1, data.r2, data.
data['splash'] = data.splash.fillna(0)
data['splash'] = [0 if x == 0 else 1 for x in data.splash]
```

In [3]:
```
data.describe()
```

```
/Users/chris/anaconda/lib/python3.5/site-packages/numpy/lib/function_b
ase.py:3834: RuntimeWarning: Invalid value encountered in percentile
  RuntimeWarning)
```

Out[3]:

	draft	splash	lands	twos	rares	x1	x2	white
count	365.000000	365.000000	365.000000	365.000000	365.000000	0.0	0.0	365.00000
mean	19.241096	0.205479	16.493151	4.876712	1.879452	NaN	NaN	0.402740
std	12.758189	0.404606	0.557745	1.931130	1.223281	NaN	NaN	0.491122
min	1.000000	0.000000	15.000000	0.000000	0.000000	NaN	NaN	0.000000
25%	8.000000	0.000000	16.000000	4.000000	1.000000	NaN	NaN	0.000000
50%	19.000000	0.000000	17.000000	5.000000	2.000000	NaN	NaN	0.000000
75%	30.000000	0.000000	17.000000	6.000000	3.000000	NaN	NaN	1.000000
max	42.000000	1.000000	18.000000	11.000000	6.000000	NaN	NaN	1.000000

In [4]:
```
#Find colorcombination win Percentages
data1 = data.groupby('color').agg({'wins':'sum', 'losses':'sum'})
data1['perc'] = data1.wins/(data1.losses+data1.wins)
data1[['wins', 'losses', 'perc']].sort_values('perc', 0, False)
```

Out[4]:

	wins	losses	perc
color			
Red	11	4	0.733333
Green	2	1	0.666667
White, Blue	53	34	0.609195
White, Red	57	42	0.575758
White, Black	71	63	0.529851
Blue, Red	41	38	0.518987
Red, Green	49	47	0.510417
Black	3	3	0.500000
Blue	1	1	0.500000
Black, Red	46	48	0.489362
Black, Green	62	67	0.480620
Blue, Green	45	53	0.459184
Blue, Black	43	56	0.434343
White, Green	37	54	0.406593
White	3	8	0.272727

In [5]:
```
print(data.groupby('draft').agg({'white':'sum', 'blue':'sum', 'black':'su
data.groupby('draft').agg({'white':'sum', 'blue':'sum', 'black':'sum', 'r
```

```
green    3.452381
red      3.190476
blue     3.071429
white    3.500000
black    3.857143
dtype: float64
```

Out[5]:

	green	red	blue	white	black
draft					

1	7	10	10	8	10
2	8	5	7	8	9
3	6	6	6	6	8
4	6	4	4	2	4
5	5	1	3	6	3
6	2	2	2	2	3
7	4	4	2	2	4
8	2	4	3	4	2
9	1	3	3	5	4
10	4	3	2	3	4
11	3	4	4	2	2
12	3	3	4	5	4
13	3	3	3	4	3
14	4	2	3	3	3
15	3	3	2	5	3
16	3	3	3	3	4
17	3	3	4	2	4
18	3	3	3	4	3
19	3	3	3	3	4
20	4	3	1	2	4
21	4	3	4	4	5
22	2	3	1	2	4
23	5	3	3	3	3
24	3	3	3	3	1
25	3	2	3	3	4
26	3	1	3	2	2
27	2	3	3	3	7
28	5	2	2	3	2
29	4	3	3	6	2

30	4	3	2	2	4
31	2	3	1	4	4
32	3	3	3	4	4
33	3	4	4	5	2
34	4	2	3	3	4
35	4	3	3	2	4
36	1	4	4	1	4
37	3	3	0	2	4
38	3	3	3	4	3
39	3	5	2	2	4
40	3	3	1	4	5
41	2	0	3	3	0
42	2	3	3	3	5

```
In [6]: data.groupby('rares').agg({'wins':'sum', 'losses':'sum'})
```

Out[6]:

	wins	**losses**
rares		
0	63	68
1	153	139
2	144	170
3	109	91
4	47	39
5	6	11
6	2	1

```
In [7]: data.groupby('wins').rares.mean()
```

```
Out[7]: wins
        0    1.896552
        1    1.781955
        2    1.954198
        3    1.930233
        Name: rares, dtype: float64
```

In [8]:
```
#Find colorcombination win Percentages
data1 = data.groupby('lands').agg({'wins':'sum', 'losses':'sum'})
data1['perc'] = data1.wins/(data1.losses+data1.wins)
data1[['wins', 'losses', 'perc']].sort_values('perc', 0, False)
```

Out[8]:

	wins	losses	perc
lands			
18	2	1	0.666667
15	16	12	0.571429
16	246	226	0.521186
17	260	280	0.481481

In [9]:
```
#Find colorcombination win Percentages
data1 = data.groupby('splash').agg({'wins':'sum', 'losses':'sum'})
data1['perc'] = data1.wins/(data1.losses+data1.wins)
data1[['wins', 'losses', 'perc']].sort_values('perc', 0, False)
```

Out[9]:

	wins	losses	perc
splash			
0	423	404	0.511487
1	101	115	0.467593

In [10]:
```
Counter(data[data['wins']==3].sort_values('draft')['color'])
```

Out[10]:
```
Counter({'Black, Green': 3,
         'Black, Red': 3,
         'Blue, Black': 4,
         'Blue, Green': 3,
         'Blue, Red': 3,
         'Red': 2,
         'Red, Green': 5,
         'White, Black': 5,
         'White, Blue': 6,
         'White, Green': 3,
         'White, Red': 6})
```

In [11]:
```
#Find colorcombination win Percentages
data1 = data.groupby('twos').agg({'wins':'sum', 'losses':'sum'})
data1['perc'] = data1.wins/(data1.losses+data1.wins)
data1[['wins', 'losses', 'perc']].sort_values('perc', 0, False)
```

Out[11]:

	wins	losses	perc
twos			
11	3	0	1.000000
7	50	32	0.609756
8	25	18	0.581395
10	12	9	0.571429
9	20	17	0.540541
4	127	122	0.510040
6	84	82	0.506024
3	74	81	0.477419
5	95	106	0.472637
1	8	12	0.400000
2	25	38	0.396825
0	1	2	0.333333

In [12]:
```
data1 = data.groupby(['c1', 'c2']).agg({'wins':'sum', 'losses':'sum'})
data1['perc'] = data1.wins/(data1.losses+data1.wins)
data1[['wins', 'losses', 'perc']]
```

Out[12]:

		wins	losses	perc
c1	**c2**			
Black		3	3	0.500000
	Green	62	67	0.480620
	Red	46	48	0.489362
Blue		1	1	0.500000
	Black	43	56	0.434343
	Green	45	53	0.459184
	Red	41	38	0.518987
Green		2	1	0.666667
Red		11	4	0.733333
	Green	49	47	0.510417
White		3	8	0.272727
	Black	71	63	0.529851
	Blue	53	34	0.609195
	Green	37	54	0.406593
	Red	57	42	0.575758

In [13]:
```
#Find colorcombination win Percentages
data1 = data.groupby('name').agg({'wins':'sum', 'losses':'sum'})
data1['perc'] = data1.wins/(data1.losses+data1.wins)
data1[['wins', 'losses', 'perc']].sort_values('perc', 0, False)
```

Out[13]:

	wins	losses	perc
name			
Eric Froehlich	3	0	1.000000
Ben Weitz	2	0	1.000000
Abe Stein	13	2	0.866667
Sam Sherman	3	1	0.750000

Stephen neal	13	5	0.722222
Pat Cox	2	1	0.666667
Pascal Maynard	16	8	0.666667
Patrick reynolds	4	2	0.666667
Siggy	10	5	0.666667
Bradley Robinson	6	3	0.666667
Alex Majlaton	15	8	0.652174
Evan Whitehouse	13	7	0.650000
Nate Sturm	7	4	0.636364
Seth Manfield	7	4	0.636364
Ben Nikolich	17	10	0.629630
Jake Mondello	15	9	0.625000
Patrick Johnson	5	3	0.625000
Timothy Wu	13	8	0.619048
Anand Khare	11	7	0.611111
Ari Lax	15	10	0.600000
Ben Anderson	12	8	0.600000
Sawyer Lucy	9	6	0.600000
Steve Rubin	9	6	0.600000
Brendan McKay	3	2	0.600000
Andrew Berke	12	8	0.600000
Bob Marshall	7	5	0.583333
Ondrej strasky	7	5	0.583333
Ryan Saxe	12	9	0.571429
Jon Stern	14	11	0.560000
Calcano	10	8	0.555556
...	...	...	...
Neeraj Shukla	9	11	0.450000
Ralph	4	5	0.444444

Owen Collier-Ridge	7	9	0.437500
Jarvis Yu	13	17	0.433333
Evan Appleton	6	8	0.428571
Adonnys Medrano	5	7	0.416667
Andrew Elenbogen	7	10	0.411765
Hayne	6	9	0.400000
Jeff Gottstein	7	11	0.388889
Lucas michaels	7	11	0.388889
Brendan Hurst	4	7	0.363636
Bobby Birmingham	5	9	0.357143
Chad uzzell	3	6	0.333333
Charles League	1	2	0.333333
Allen Sun	1	2	0.333333
Tony Hopkins	2	4	0.333333
Rachel Otto	6	14	0.300000
Josh Cho	3	7	0.300000
Adam Ragsdale	6	14	0.300000
Chas Hinkle	2	6	0.250000
Daniel Zeffert	3	12	0.200000
Lance Hartbarger	2	9	0.181818
Jamie Miller	1	5	0.166667
Liam Bollard	1	9	0.100000
Steven Mccoy	0	4	0.000000
Mark Jacobson	0	2	0.000000
Paul Yeem	0	2	0.000000
Kyle wickenheiser	0	6	0.000000
Naod Haddish	0	2	0.000000
35	0	3	0.000000

82 rows × 3 columns

```
In [ ]:
```