visualization

July 13, 2019

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        import warnings
        warnings.filterwarnings("ignore")

In [2]: train = pd.read_csv('./train_V2.csv')

In [3]: train.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4446966 entries, 0 to 4446965
Data columns (total 29 columns):
Id                 object
groupId            object
matchId            object
assists            int64
boosts             int64
damageDealt        float64
DBNOs              int64
headshotKills      int64
heals              int64
killPlace          int64
killPoints         int64
kills              int64
killStreaks        int64
longestKill        float64
matchDuration      int64
matchType          object
maxPlace           int64
numGroups          int64
rankPoints         int64
revives            int64
rideDistance       float64
roadKills          int64
swimDistance       float64
teamKills          int64
```

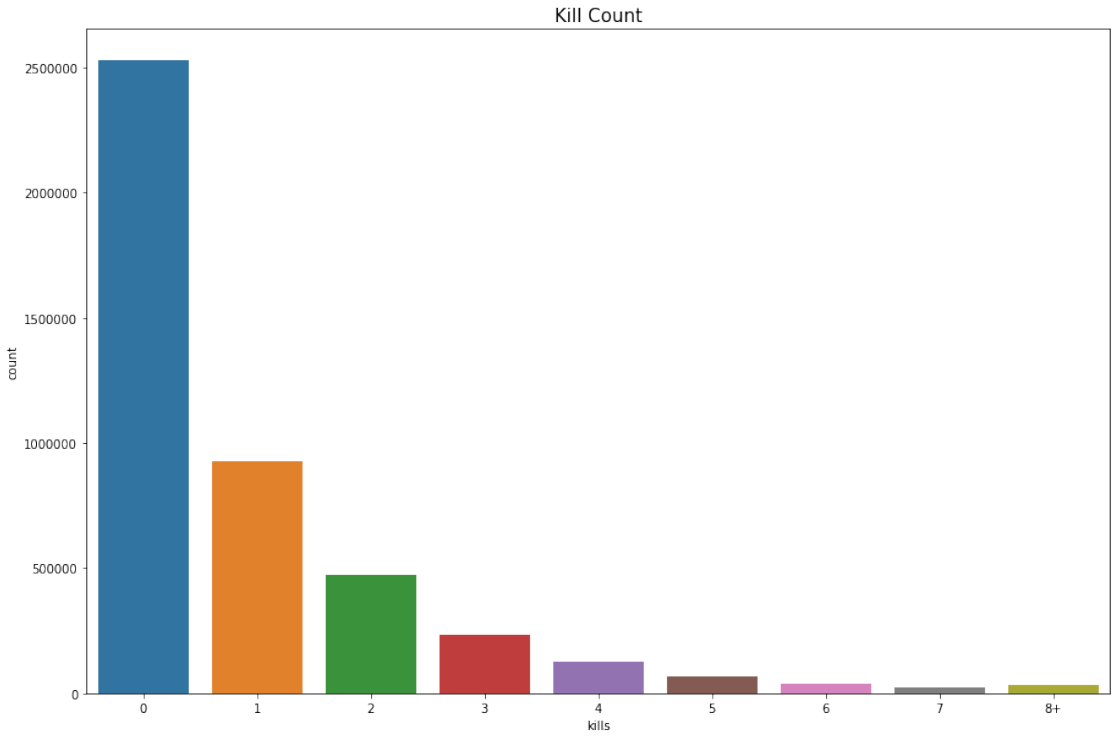
```
vehicleDestroys     int64
walkDistance        float64
weaponsAcquired     int64
winPoints           int64
winPlacePerc        float64
dtypes: float64(6), int64(19), object(4)
memory usage: 983.9+ MB
```

0.1 Kill

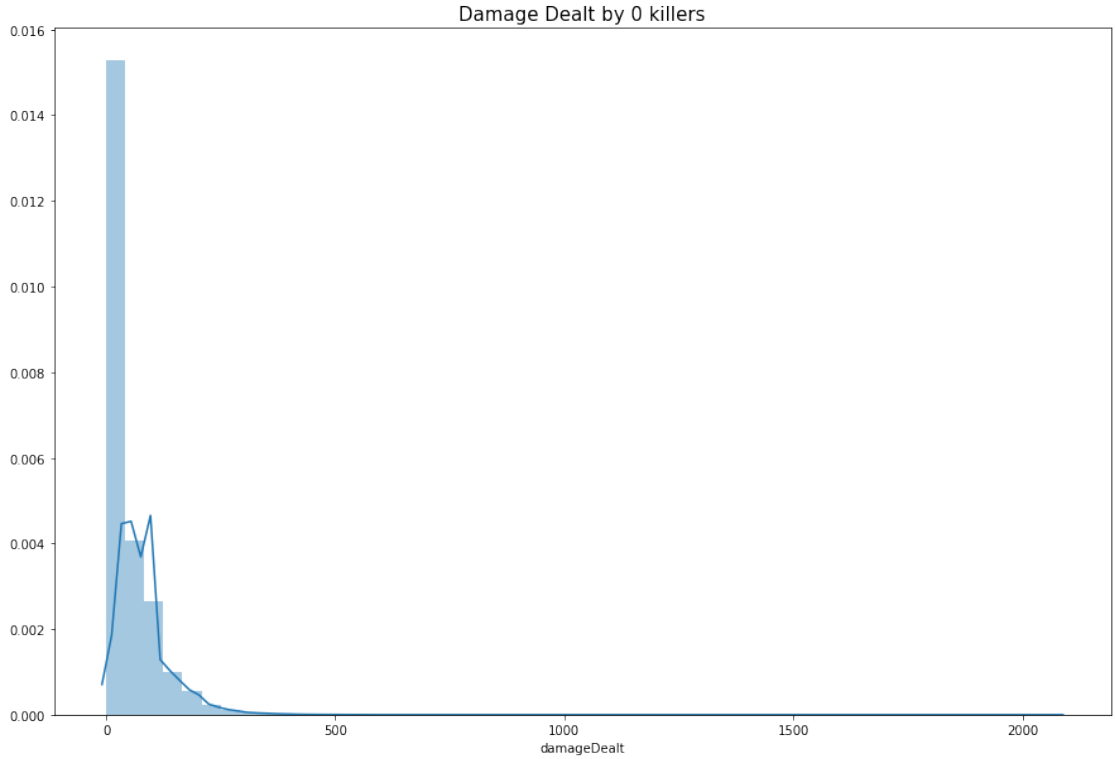
```
In [7]: print("The average person kills {:.4f} players, 99% of people have {} kills or less, whi
            train['kills'].mean(),train['kills'].quantile(0.99), train['kills'].max()))
```

```
The average person kills 0.9248 players, 99% of people have 7.0 kills or less, while the most ki
```

```
In [5]: data = train.copy()
        data.loc[data['kills'] > data['kills'].quantile(0.99)] = '8+'
        plt.figure(figsize=(15,10))
        sns.countplot(data['kills'].astype('str').sort_values())
        plt.title("Kill Count",fontsize=15)
        plt.show()
```



```
In [6]: data = train.copy()
        data = data[data['kills']==0]
        plt.figure(figsize=(15,10))
        plt.title("Damage Dealt by 0 killers",fontsize=15)
        sns.distplot(data['damageDealt'])
        plt.show()
```

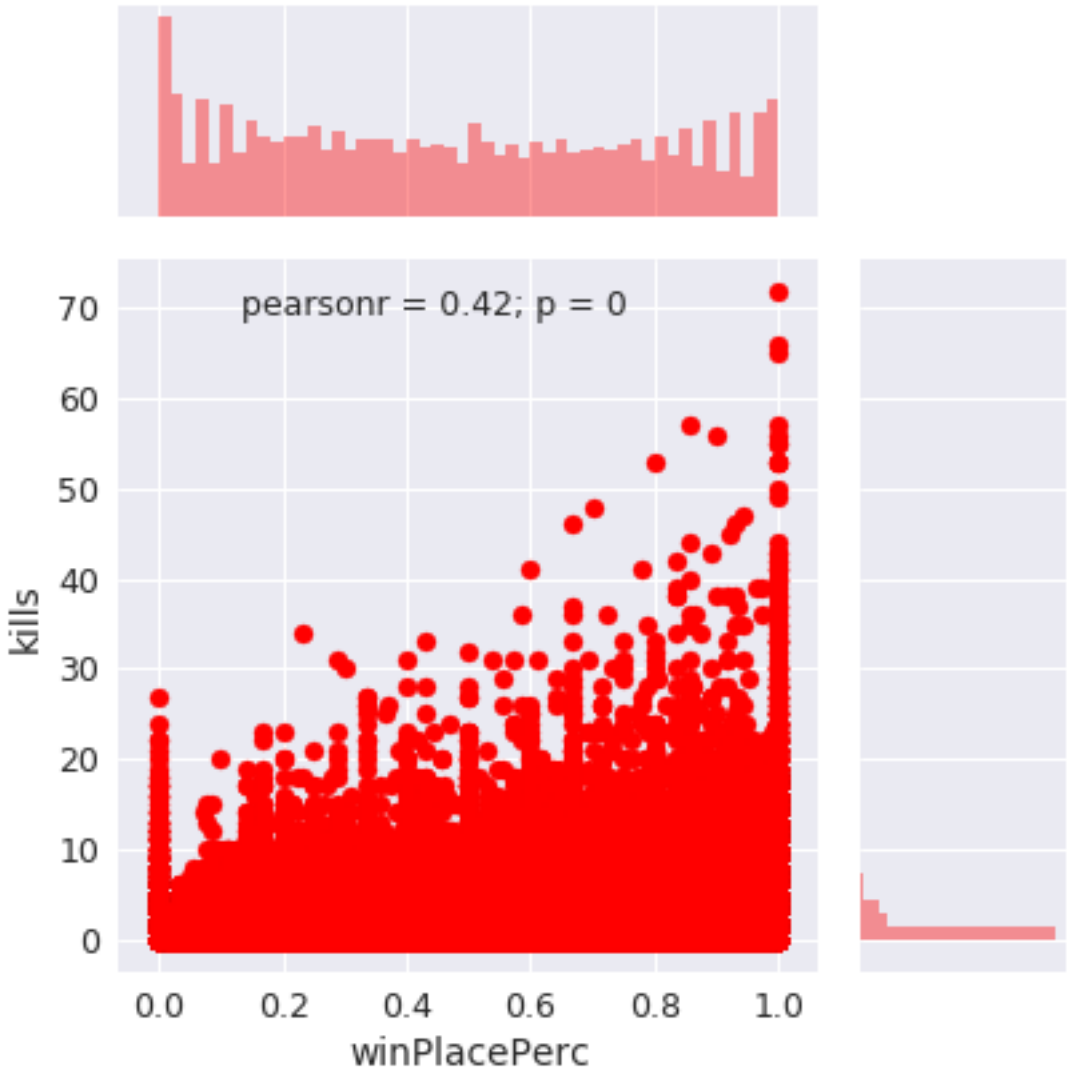



```
In [8]: print("{} players ({:.4f}%) have won without a single kill!".format(len(data[data['winPl

        data1 = train[train['damageDealt'] == 0].copy()
        print("{} players ({:.4f}%) have won without dealing damage!".format(len(data1[data1['wi

16666 players (0.3748%) have won without a single kill!
4770 players (0.1073%) have won without dealing damage!
```

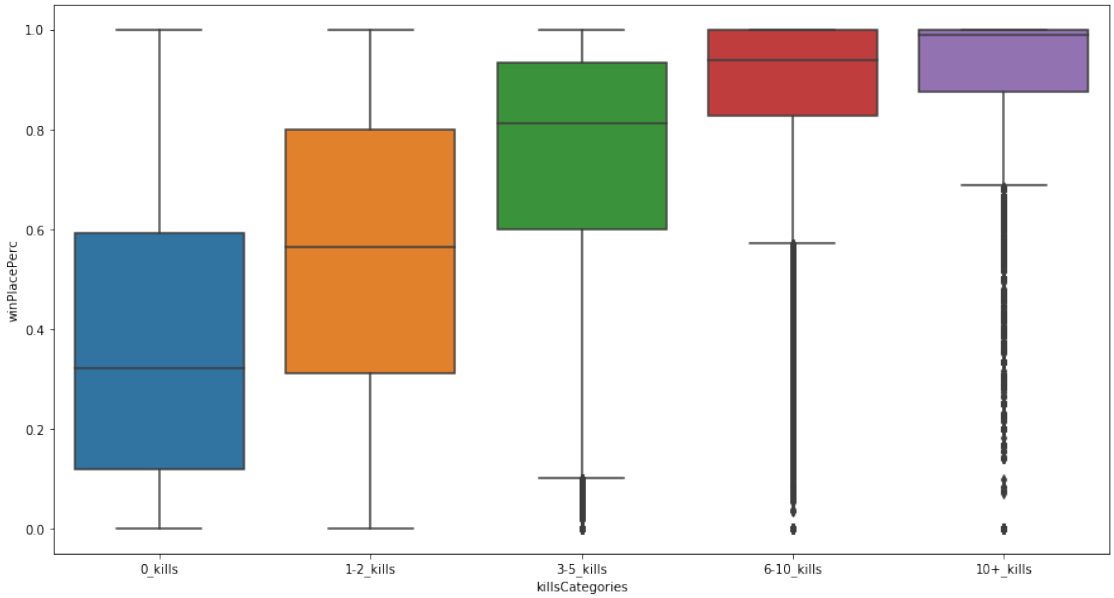
```
In [31]: sns.jointplot(x="winPlacePerc", y="kills", data=train, ratio=3, color="r")
         plt.show()
```




```
In [10]: kills = train.copy()

         kills['killsCategories'] = pd.cut(kills['kills'], [-1, 0, 2, 5, 10, 60], labels=['0_kil

         plt.figure(figsize=(15,8))
         sns.boxplot(x="killsCategories", y="winPlacePerc", data=kills)
         plt.show()
```

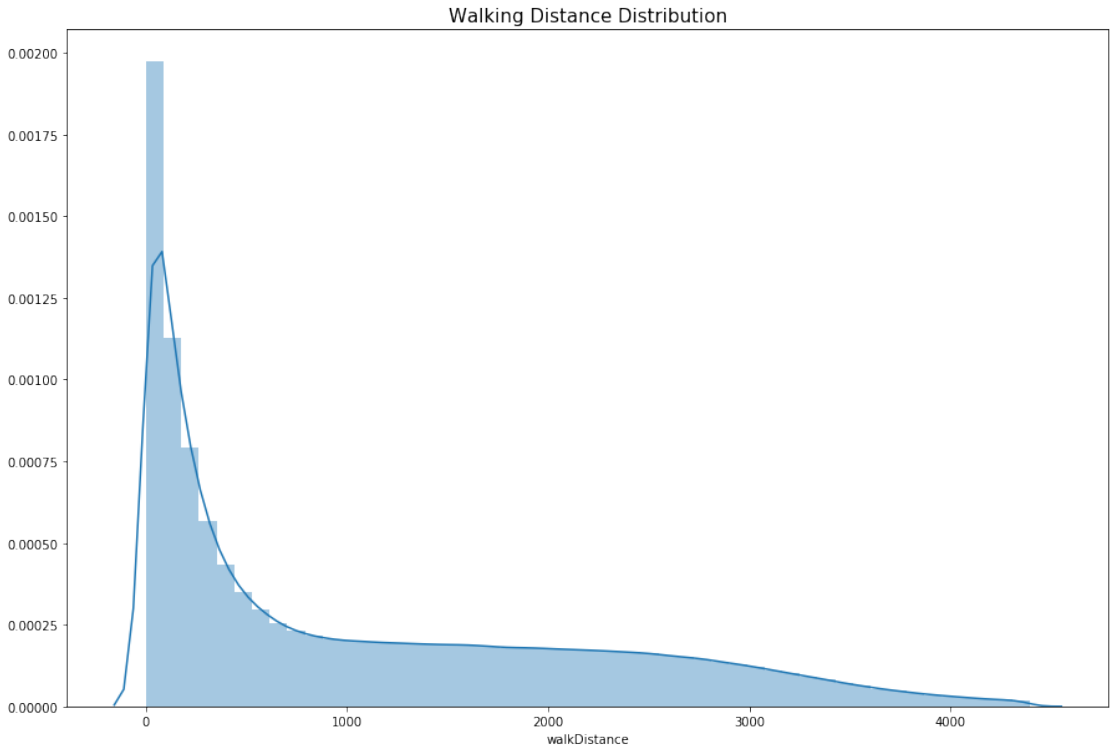


0.2 Walking distance

```
In [12]: print("The average person walks for {:.1f}m, 99% of people have walked {}m or less, whi
             train['walkDistance'].mean(), train['walkDistance'].quantile(0.99), train['walkDist
```

```
The average person walks for 1154.2m, 99% of people have walked 4396.0m or less, while the marat
```

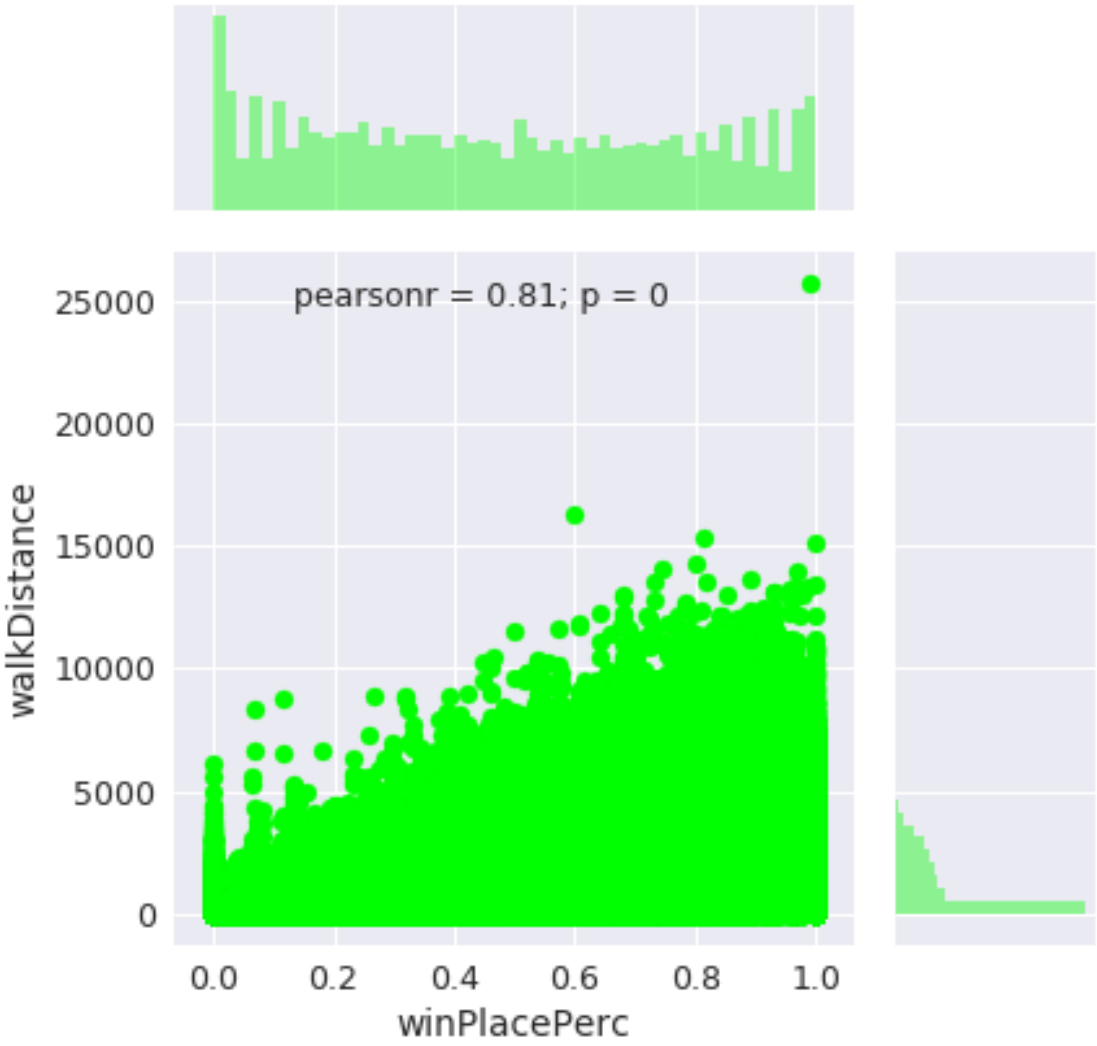
```
In [11]: data = train.copy()
         data = data[data['walkDistance'] < train['walkDistance'].quantile(0.99)]
         plt.figure(figsize=(15,10))
         plt.title("Walking Distance Distribution",fontsize=15)
         sns.distplot(data['walkDistance'])
         plt.show()
```




```
In [13]: print("{} players ({:.4f}%) walked 0 meters. This means that they die before even takin
```

```
99603 players (2.0329%) walked 0 meters. This means that they die before even taking a step or t
```

```
In [30]: sns.jointplot(x="winPlacePerc", y="walkDistance",  data=train, ratio=3, color="lime")
         plt.show()
```



0.3 Heal and Boost

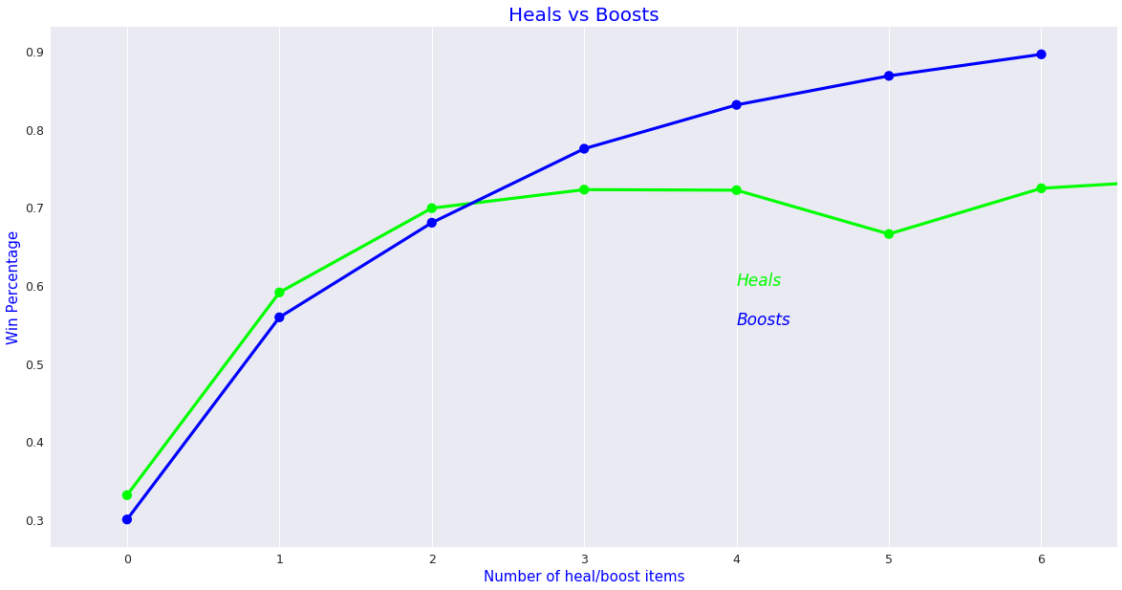
```
In [27]: print("The average person uses {:.1f} heal items, 99% of people use {} or less, while t
         print("The average person uses {:.1f} boost items, 99% of people use {} or less, while
```

```
The average person uses 1.4 heal items, 99% of people use 12.0 or less, while the doctor used 80
The average person uses 1.1 boost items, 99% of people use 7.0 or less, while the doctor used 33
```

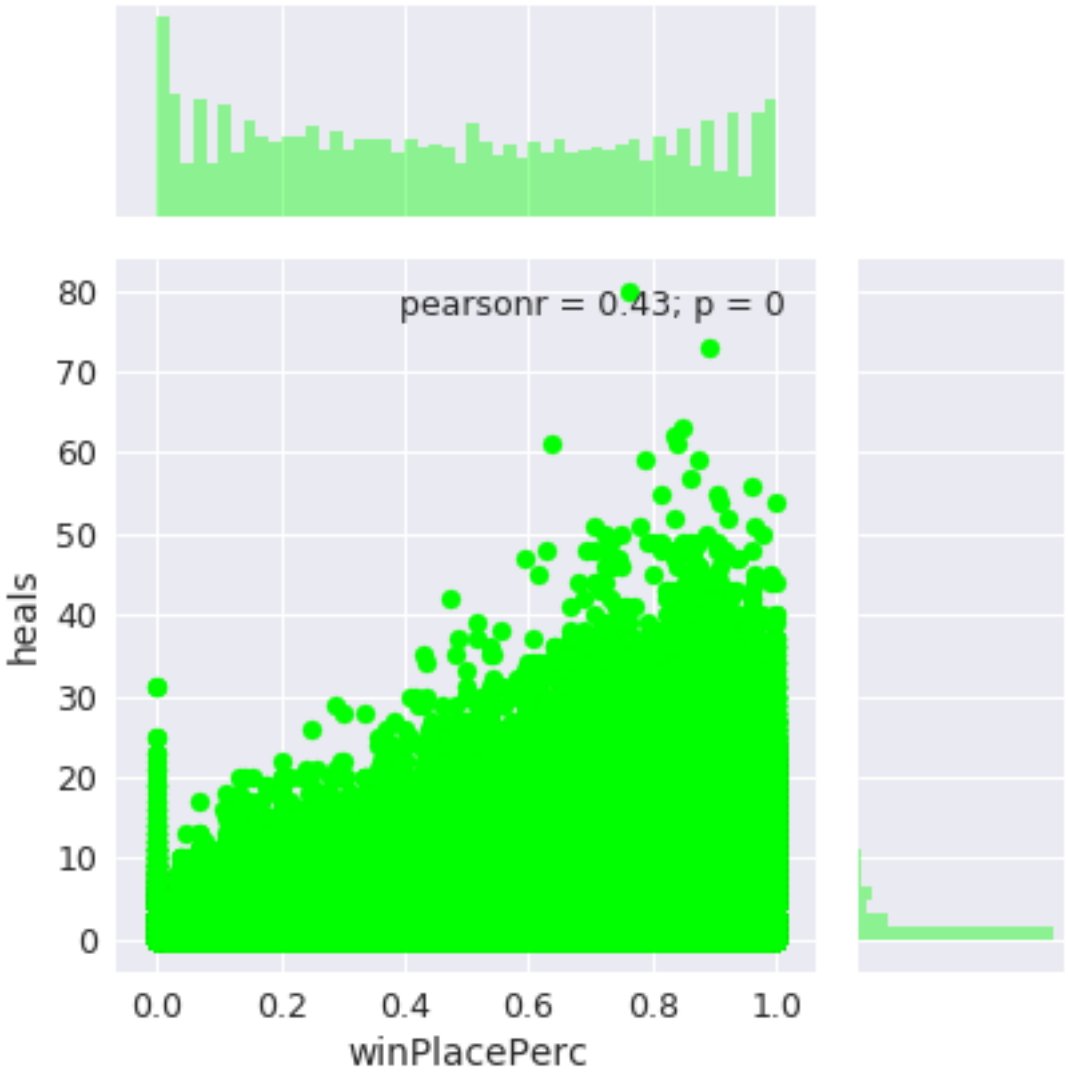
```
In [28]: data = train.copy()
         data = data[data['heals'] < data['heals'].quantile(0.99)]
         data = data[data['boosts'] < data['boosts'].quantile(0.99)]

         f,ax1 = plt.subplots(figsize =(20,10))
         sns.pointplot(x='heals',y='winPlacePerc',data=data,color='lime',alpha=0.8)
```

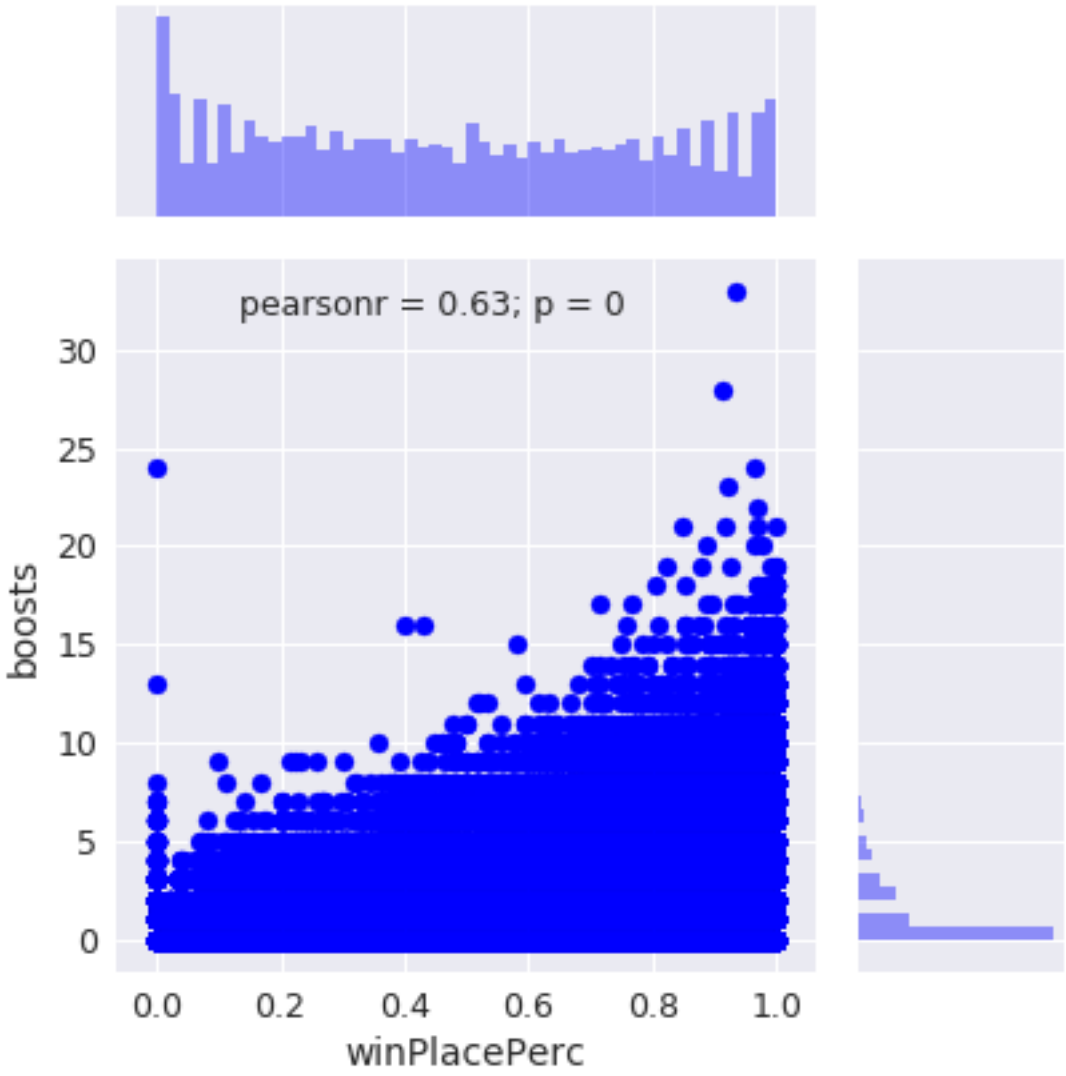
```
sns.pointplot(x='boosts',y='winPlacePerc',data=data,color='blue',alpha=0.8)
plt.text(4,0.6,'Heals',color='lime',fontsize = 17,style = 'italic')
plt.text(4,0.55,'Boosts',color='blue',fontsize = 17,style = 'italic')
plt.xlabel('Number of heal/boost items',fontsize = 15,color='blue')
plt.ylabel('Win Percentage',fontsize = 15,color='blue')
plt.title('Heals vs Boosts',fontsize = 20,color='blue')
plt.grid()
plt.show()
```



In [33]:
```
sns.jointplot(x="winPlacePerc", y="heals", data=train, ratio=3, color="lime")
plt.show()
```




```
In [34]: sns.jointplot(x="winPlacePerc", y="boosts", data=train, ratio=3, color="blue")
         plt.show()
```



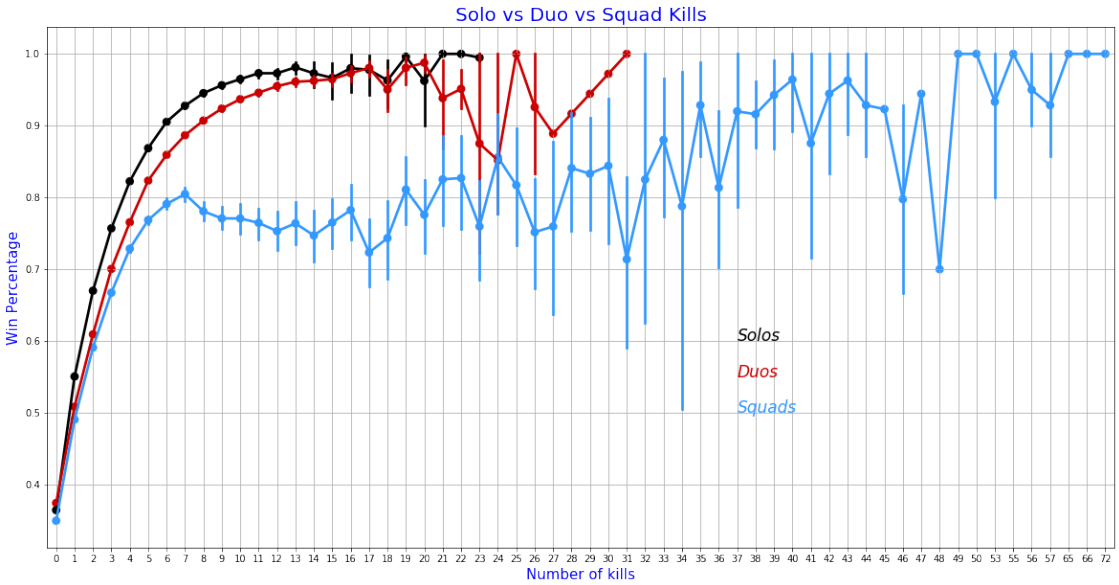
0.4 Solos, Duos and Squads

```
In [15]: solos = train[train['numGroups']>50]
         duos = train[(train['numGroups']>25) & (train['numGroups']<=50)]
         squads = train[train['numGroups']<=25]
         print("There are {} ({:.2f}%) solo games, {} ({:.2f}%) duo games and {} ({:.2f}%) squad
```

```
There are 709111 (15.95%) solo games, 3295326 (74.10%) duo games and 442529 (9.95%) squad games.
```

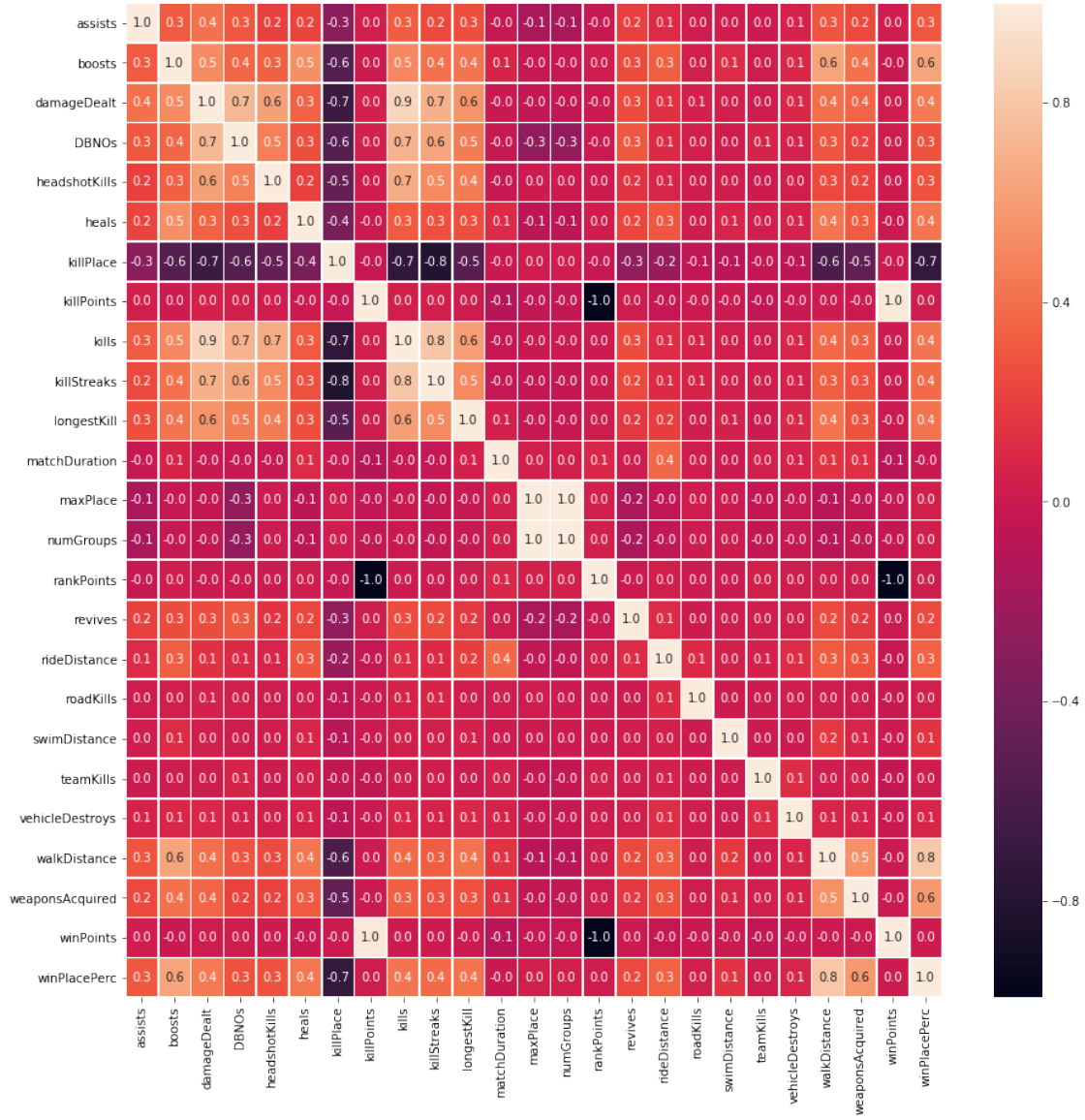
```
In [16]: f,ax1 = plt.subplots(figsize =(20,10))
         sns.pointplot(x='kills',y='winPlacePerc',data=solos,color='black',alpha=0.8)
         sns.pointplot(x='kills',y='winPlacePerc',data=duos,color='#CC0000',alpha=0.8)
         sns.pointplot(x='kills',y='winPlacePerc',data=squads,color='#3399FF',alpha=0.8)
```

```
plt.text(37,0.6,'Solos',color='black',fontsize = 17,style = 'italic')
plt.text(37,0.55,'Duos',color='#CC0000',fontsize = 17,style = 'italic')
plt.text(37,0.5,'Squads',color='#3399FF',fontsize = 17,style = 'italic')
plt.xlabel('Number of kills',fontsize = 15,color='blue')
plt.ylabel('Win Percentage',fontsize = 15,color='blue')
plt.title('Solo vs Duo vs Squad Kills',fontsize = 20,color='blue')
plt.grid()
plt.show()
```

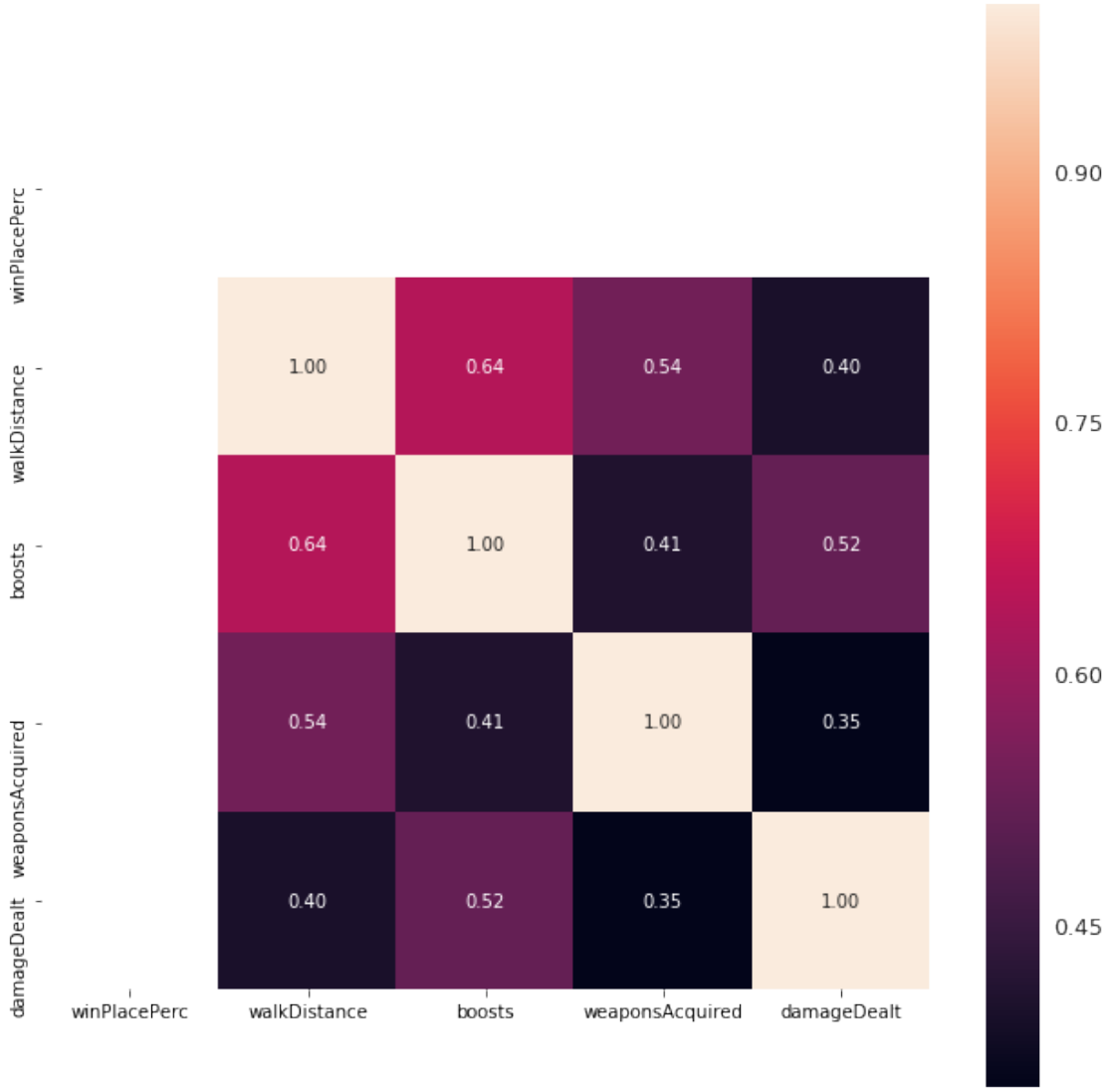


0.5 Pearson correlation between variables

In [18]:
```
f,ax = plt.subplots(figsize=(15, 15))
sns.heatmap(train.corr(), annot=True, linewidths=.5, fmt= '.1f',ax=ax)
plt.show()
```



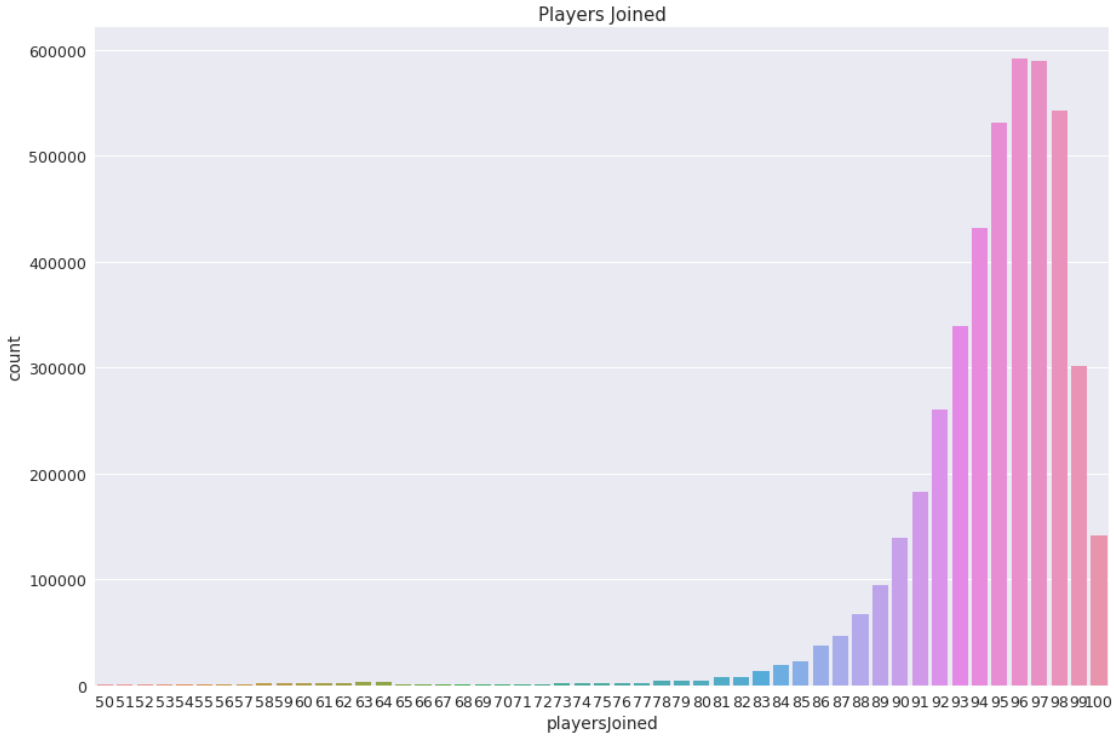

```
In [19]: k = 5 #number of variables for heatmap
         f,ax = plt.subplots(figsize=(11, 11))
         cols = train.corr().nlargest(k, 'winPlacePerc')['winPlacePerc'].index
         cm = np.corrcoef(train[cols].values.T)
         sns.set(font_scale=1.25)
         hm = sns.heatmap(cm, cbar=True, annot=True, square=True, fmt='.2f', annot_kws={'size':
         plt.show()
```



0.6 Feature Engineering

```
In [20]: train['playersJoined'] = train.groupby('matchId')['matchId'].transform('count')
```

```
In [21]: data = train.copy()
         data = data[data['playersJoined']>49]
         plt.figure(figsize=(15,10))
         sns.countplot(data['playersJoined'])
         plt.title("Players Joined",fontsize=15)
         plt.show()
```




```
In [22]: train['killsNorm'] = train['kills']*((100-train['playersJoined'])/100 + 1)
         train['damageDealtNorm'] = train['damageDealt']*((100-train['playersJoined'])/100 + 1)
         train[['playersJoined', 'kills', 'killsNorm', 'damageDealt', 'damageDealtNorm']][5:8]
```

Out[22]:

	playersJoined	kills	killsNorm	damageDealt	damageDealtNorm
5	95	1	1.05	100.000	105.00000
6	97	0	0.00	0.000	0.00000
7	96	0	0.00	8.538	8.87952

```
In [23]: train['healsAndBoosts'] = train['heals']+train['boosts']
         train['totalDistance'] = train['walkDistance']+train['rideDistance']+train['swimDistanc
```

```
In [24]: train['boostsPerWalkDistance'] = train['boosts']/(train['walkDistance']+1) #The +1 is t
         train['boostsPerWalkDistance'].fillna(0, inplace=True)
         train['healsPerWalkDistance'] = train['heals']/(train['walkDistance']+1) #The +1 is to
         train['healsPerWalkDistance'].fillna(0, inplace=True)
         train['healsAndBoostsPerWalkDistance'] = train['healsAndBoosts']/(train['walkDistance']
         train['healsAndBoostsPerWalkDistance'].fillna(0, inplace=True)
         train[['walkDistance', 'boosts', 'boostsPerWalkDistance' ,'heals',  'healsPerWalkDistan
```

Out[24]:

	walkDistance	boosts	boostsPerWalkDistance	heals	healsPerWalkDistance	\
40	327.30	1	0.003046	1	0.003046	
41	128.80	0	0.000000	0	0.000000	
42	52.52	0	0.000000	0	0.000000	

```
43        534.10       1               0.001869      0              0.000000
44       2576.00       4               0.001552      6              0.002328

    healsAndBoosts  healsAndBoostsPerWalkDistance
40               2                       0.006092
41               0                       0.000000
42               0                       0.000000
43               1                       0.001869
44              10                       0.003880
```

In [25]:
```
train['killsPerWalkDistance'] = train['kills']/(train['walkDistance']+1) #The +1 is to
train['killsPerWalkDistance'].fillna(0, inplace=True)
train[['kills', 'walkDistance', 'rideDistance', 'killsPerWalkDistance', 'winPlacePerc']
```

Out[25]:

	kills	walkDistance	rideDistance	killsPerWalkDistance	winPlacePerc
4115816	29	0.0	0.0	29.0	0.7500
3083358	30	0.0	0.0	30.0	0.7500
422093	30	0.0	0.0	30.0	1.0000
2394021	31	0.0	0.0	31.0	0.5385
3057746	31	0.0	0.0	31.0	0.7500
2998470	35	0.0	0.0	35.0	1.0000
1158891	36	0.0	0.0	36.0	0.5833
3062788	36	0.0	0.0	36.0	0.8667
1068513	38	0.0	0.0	38.0	0.8333
1702541	43	0.0	0.0	43.0	1.0000

In [26]:
```
train['team'] = [1 if i>50 else 2 if (i>25 & i<=50) else 4 for i in train['numGroups']]
train.head()
```

Out[26]:

	Id	groupId	matchId	assists	boosts	\
0	7f96b2f878858a	4d4b580de459be	a10357fd1a4a91	0	0	
1	eef90569b9d03c	684d5656442f9e	aeb375fc57110c	0	0	
2	1eaf90ac73de72	6a4a42c3245a74	110163d8bb94ae	1	0	
3	4616d365dd2853	a930a9c79cd721	f1f1f4ef412d7e	0	0	
4	315c96c26c9aac	de04010b3458dd	6dc8ff871e21e6	0	0	

	damageDealt	DBNOs	headshotKills	heals	killPlace	...	playersJoined	\
0	0.00	0	0	0	60	...	96	
1	91.47	0	0	0	57	...	91	
2	68.00	0	0	0	47	...	98	
3	32.90	0	0	0	75	...	91	
4	100.00	0	0	0	45	...	97	

	killsNorm	damageDealtNorm	healsAndBoosts	totalDistance	\
0	0.00	0.0000	0	244.8000	
1	0.00	99.7023	0	1445.0445	
2	0.00	69.3600	0	161.8000	
3	0.00	35.8610	0	202.7000	
4	1.03	103.0000	0	49.7500	

```
   boostsPerWalkDistance  healsPerWalkDistance  healsAndBoostsPerWalkDistance  \
0                    0.0                   0.0                            0.0
1                    0.0                   0.0                            0.0
2                    0.0                   0.0                            0.0
3                    0.0                   0.0                            0.0
4                    0.0                   0.0                            0.0

    killsPerWalkDistance  team
0               0.000000     2
1               0.000000     4
2               0.000000     2
3               0.000000     2
4               0.019704     1

[5 rows x 39 columns]
```