**Sinan Burak Öztürk**
**213012111**

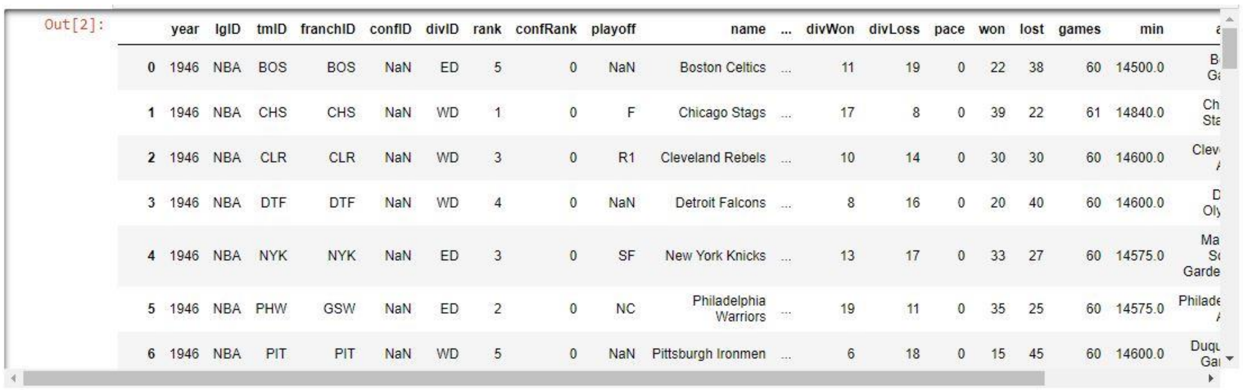
# CS 240 – Project

Questions that I can analyze and find some answers with the data:

**1)** Is there a relationship between the positions in final standings and wins in NBA League and after 1962?
**2)** Is there a relationship between winning trend between first and other years?
**3)** Is players' salaries increase as time goes on?

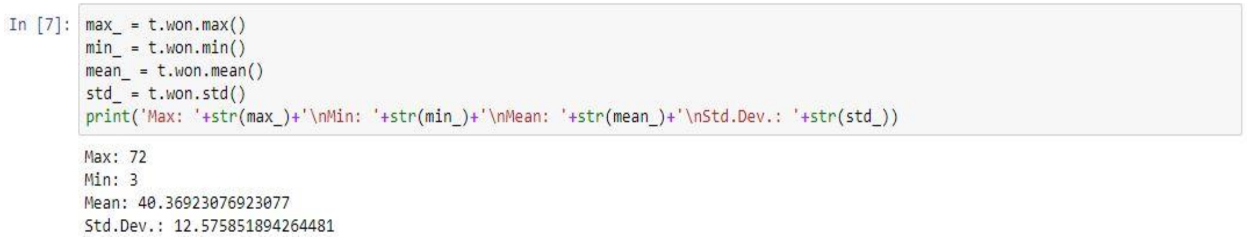
First question was my choice to analyze. I read "basketball_teams.csv" file with csv reader in Jupiter. In the csv file the values are between 1937 and 2011. However, in this time period there are 5 different leagues. ABL1, NPBL, NBL, PBLA and NBA are the leagues. I chose NBA and league years bigger than 1962 since there are at least 2 different leagues before 1963. First, I plot a histogram of the number of victories of the teams. Then, I made PMF and CDF tables of the wins. To be able to see the relationship between position in final standings and wins after 1962 I used a correlation function from the book. I was expecting negative correlation because while the number of wins increase, the number in the league decrease. In the league table, the ordering starts from 1 but the team in the first position of the league has more victory than the team at the last position of the league. After find the correlation coefficient I made a scatter plot to see the relationship between two variables clearly. To make the chart clearer I wrote jittering function and I added random noise to the chart. After jittering my chart, I examine the shape of the distribution. After that I made hypothesis testing to see if it happened by chance or the effect of it is real.

**1)** First, I read "basketball_teams.csv" file with csv reader in Jupiter notebook. The table I got first has 60 columns and 1536 rows. However, I only needed data after 1962 and NBA league. The number of rows became 1235 after I specialize the data.


Out[2]:

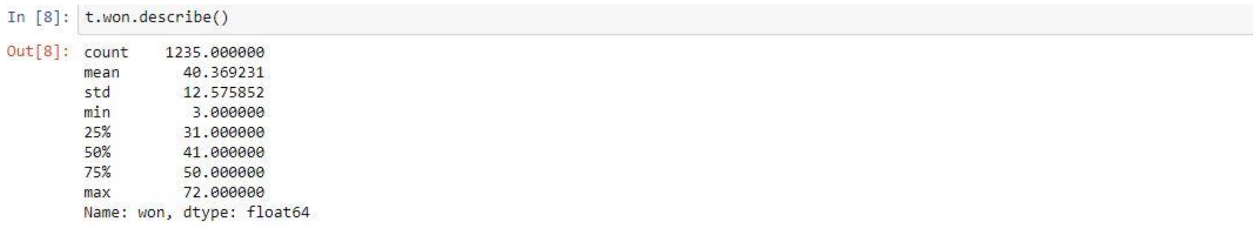
| | year | lgID | tmID | franchID | confID | divID | rank | confRank | playoff | name | ... | divWon | divLoss | pace | won | lost | games | min | a |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1946 | NBA | BOS | BOS | NaN | ED | 5 | 0 | NaN | Boston Celtics | ... | 11 | 19 | 0 | 22 | 38 | 60 | 14500.0 | B Ga |
| 1 | 1946 | NBA | CHS | CHS | NaN | WD | 1 | 0 | F | Chicago Stags | ... | 17 | 8 | 0 | 39 | 22 | 61 | 14840.0 | Ch Sta |
| 2 | 1946 | NBA | CLR | CLR | NaN | WD | 3 | 0 | R1 | Cleveland Rebels | ... | 10 | 14 | 0 | 30 | 30 | 60 | 14600.0 | Clev A |
| 3 | 1946 | NBA | DTF | DTF | NaN | WD | 4 | 0 | NaN | Detroit Falcons | ... | 8 | 16 | 0 | 20 | 40 | 60 | 14600.0 | D Oly |
| 4 | 1946 | NBA | NYK | NYK | NaN | ED | 3 | 0 | SF | New York Knicks | ... | 13 | 17 | 0 | 33 | 27 | 60 | 14575.0 | Ma Sc Garde |
| 5 | 1946 | NBA | PHW | GSW | NaN | ED | 2 | 0 | NC | Philadelphia Warriors | ... | 19 | 11 | 0 | 35 | 25 | 60 | 14575.0 | Philade A |
| 6 | 1946 | NBA | PIT | PIT | NaN | WD | 5 | 0 | NaN | Pittsburgh Ironmen | ... | 6 | 18 | 0 | 15 | 45 | 60 | 14600.0 | Duqu Gai |

**2)** I will use rank which is the position in final standings and won which is the number of victory columns in the table.



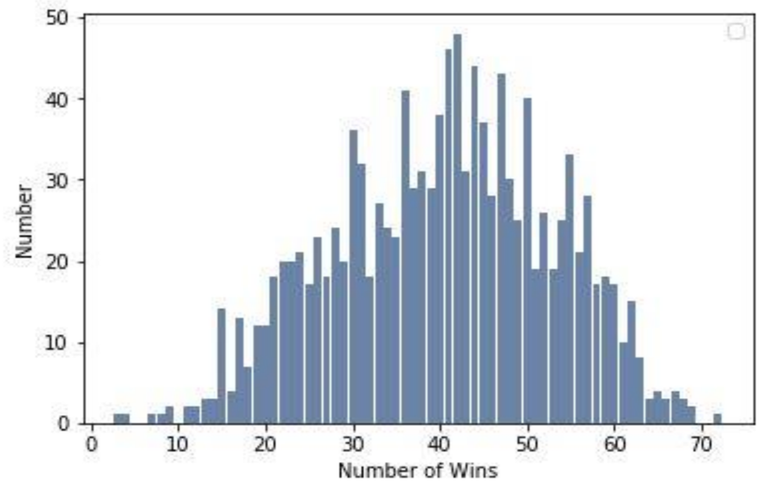
```
In [7]: max_ = t.won.max()
        min_ = t.won.min()
        mean_ = t.won.mean()
        std_ = t.won.std()
        print('Max: '+str(max_)+'\nMin: '+str(min_)+'\nMean: '+str(mean_)+'\nStd.Dev.: '+str(std_))

        Max: 72
        Min: 3
        Mean: 40.36923076923077
        Std.Dev.: 12.575851894264481
```

**3)** I calculated some statistics about the won column. The maximum number for the winning games is 72. The minimum number is 3. The mean of that column is 40.36 which means the average of the winning numbers. Standard deviation is 12.57.

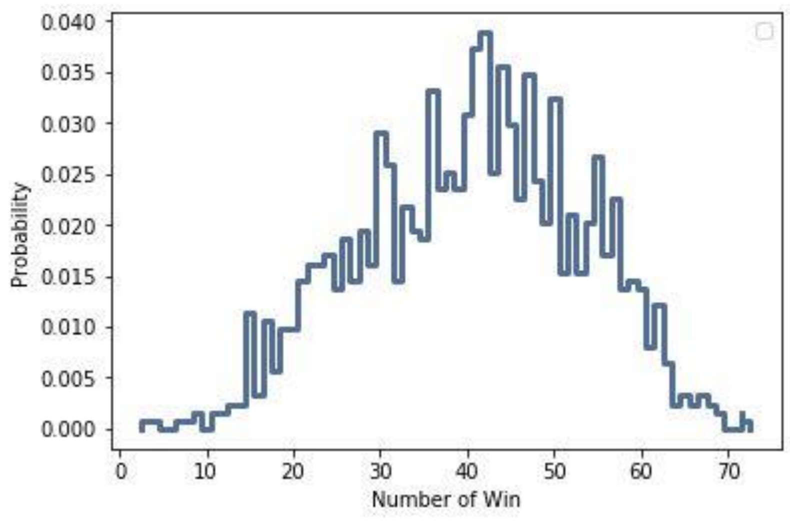


```
In [8]: t.won.describe()

Out[8]: count    1235.000000
        mean       40.369231
        std        12.575852
        min         3.000000
        25%        31.000000
        50%        41.000000
        75%        50.000000
        max        72.000000
        Name: won, dtype: float64
```

**4)** I made a histogram for the number of wins. From the histogram chart we can understand that the most number of game winning is between 40 and 50.

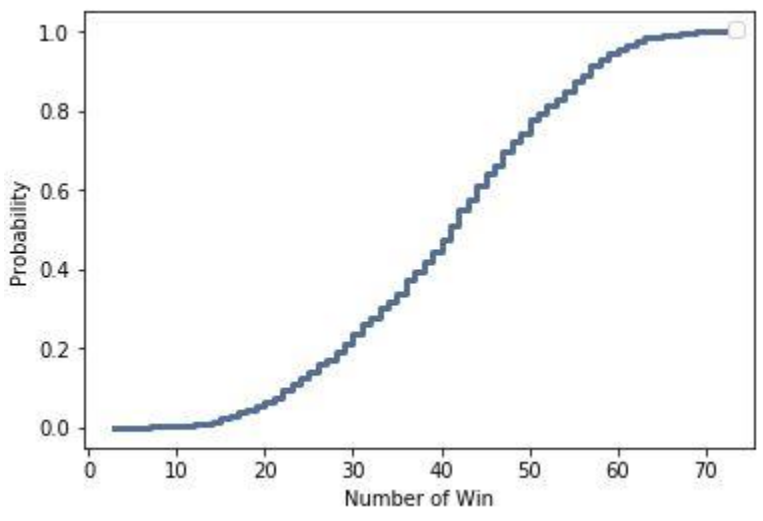


**5)** Then I made a Probability Mass Function plot for the number of wins. Probability Mass Function tell us about the probability of the number of wins happen. As we see in the histogram the most happening game winning number is between 40 and 50. From PMF, we can see that the highest probability of the game winning number is again between 40 and 50 which is higher than 0.035 but less than 0.040

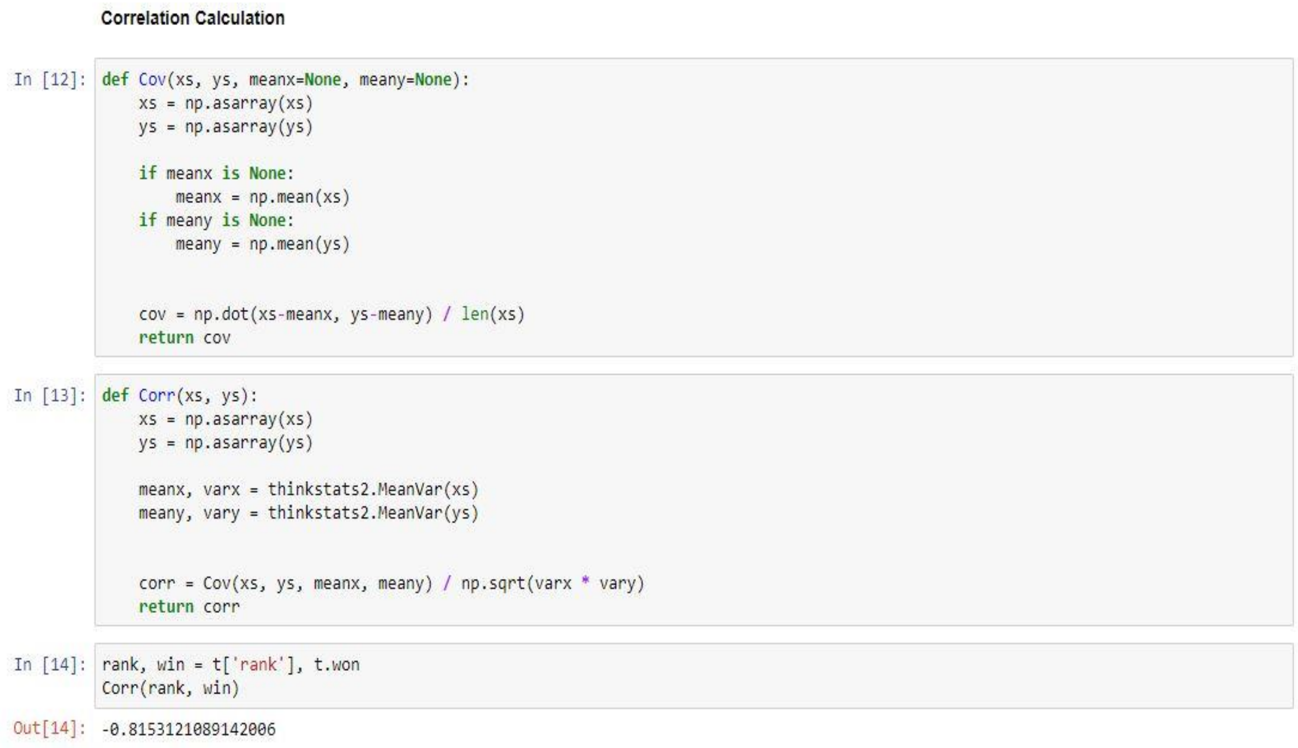


.

**6)** After that I made a cumulative distribution function plot which is starting from 0 and goes to 1 for the number of wins.



**7)** To be able to see the relationship between the number of the win and rank I used a correlation function. The correlation between winning games and rank is -0.815. Now, what we will understand from this result? As I mentioned above, I expected a negative correlation because we expect that when number of winning game increase the position in the table is decrease. We need to be careful about one key thing. The position is increase but the number of position is decrease. As we know, positions start from 1. That’s why position 1 refers the most successful team. This correlation shows us when the number of the game winning increase the number of position decrease. So that there is an opposite relationship between rank and win. In the code part we are using covariance to calculate the correlation. We subtract the means from the values and divide in to length of the values. After we found that we divide this to the square root of the multiplication of the variances. This gave us the correlation coefficient.



**Correlation Calculation**

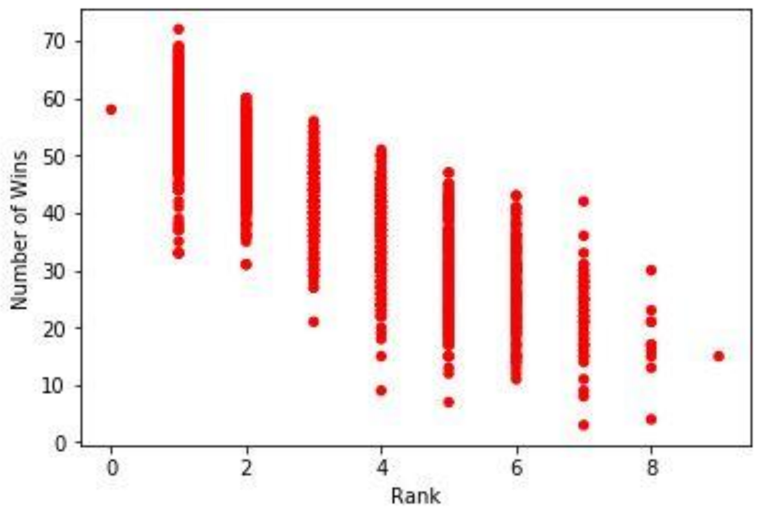
```
In [12]: def Cov(xs, ys, meanx=None, meany=None):
             xs = np.asarray(xs)
             ys = np.asarray(ys)

             if meanx is None:
                 meanx = np.mean(xs)
             if meany is None:
                 meany = np.mean(ys)


             cov = np.dot(xs-meanx, ys-meany) / len(xs)
             return cov
```

```
In [13]: def Corr(xs, ys):
             xs = np.asarray(xs)
             ys = np.asarray(ys)

             meanx, varx = thinkstats2.MeanVar(xs)
             meany, vary = thinkstats2.MeanVar(ys)


             corr = Cov(xs, ys, meanx, meany) / np.sqrt(varx * vary)
             return corr
```

```
In [14]: rank, win = t['rank'], t.won
         Corr(rank, win)

Out[14]: -0.8153121089142006
```

**8)** To see this relationship better I visualized the correlation with scatter plot. We are using scatter plot to see the relationship between two variables. As we can understand from the table the shape of the distribution in scatter plot is like decreasing. I was expecting a distribution like that because our correlation coefficient was negative.

```
thinkplot.Scatter(rank, win, alpha=1, color='Red')
thinkplot.Show(xlabel='Rank', ylabel='Number of Wins', legend=False)
```

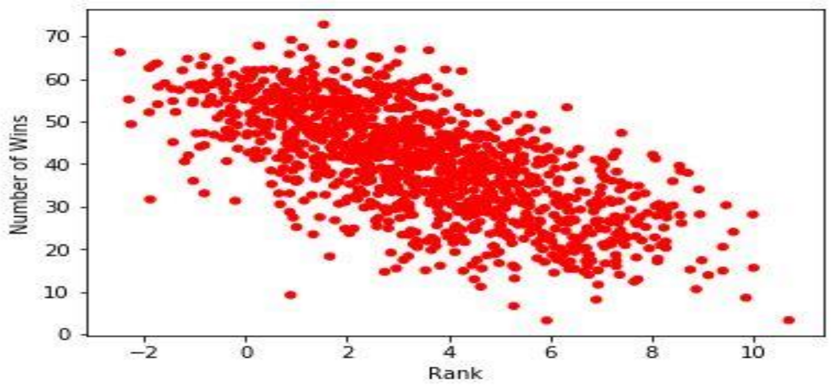


**9)** To be able to see and interpret better I used jittering. I add random noise to the distribution. After jittering the shape of the distribution became clear. Even we don't know the correlation coefficient we can understand from this chart the correlation is negative.

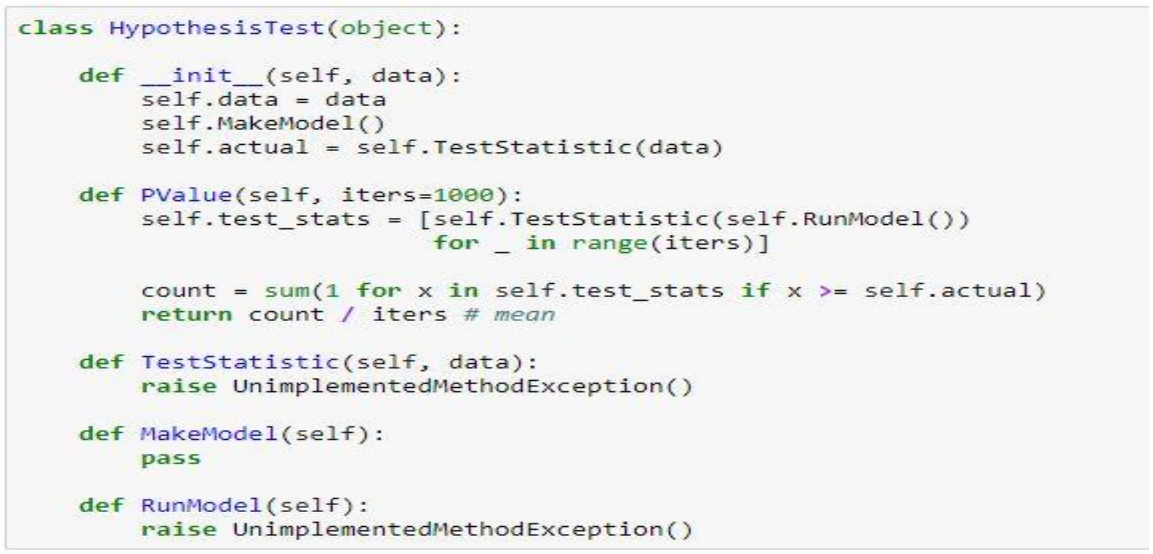
```
def Jitter(values, jitter=0.5):
    n = len(values)
    return np.random.normal(0, jitter, n) + values
```

```
rank_J = Jitter(rank, 1.4)
win_J = Jitter(win, 0.5)
```

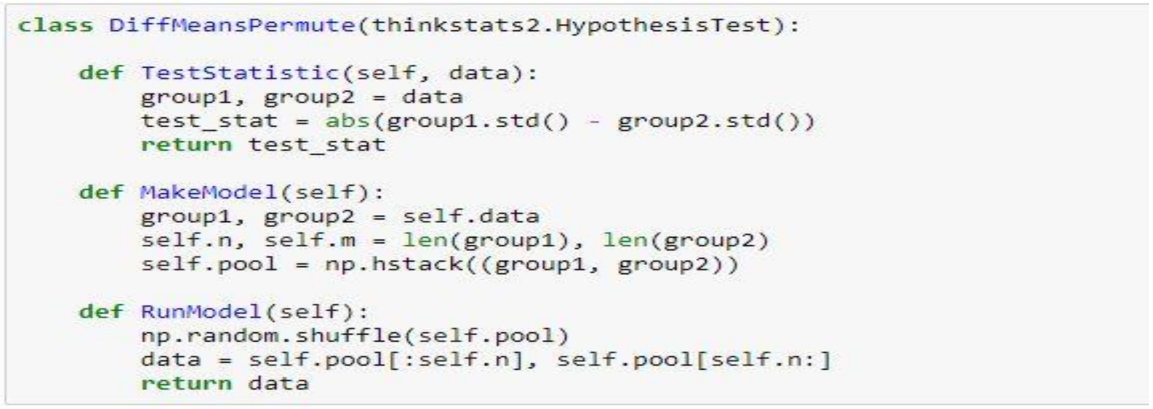
```
thinkplot.Scatter(rank_J, win_J, alpha=1, color='Red')
thinkplot.Show(xlabel='Rank', ylabel='Number of Wins', legend=False)
```



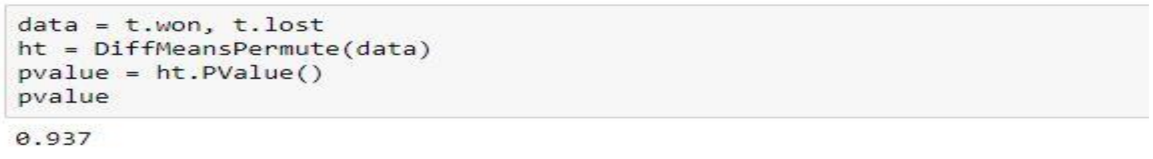
**10)** The last step is hypothesis testing. I tried to answer the question that if this affects happened by chance or not. It makes our hypothesis more accurate. In the code part, we have Pvalue. We will look to Pvalue and try to understand if we should reject our null hypothesis or not. Generally, we have 0.05 or 5% threshold. If our Pvalue smaller than threshold then it is statistically significant and we can reject our null hypothesis. After I used HypothesisTest class and another class that has a name DiffMeansPermute from the book. This class has heritage from the HypothesisTest class. I used this class to see calculate the Pvalue. In test statistic part I used the absolute differences between the standard deviation of the groups. In the code absolute is used that's why it will be one sided test. In the make model part sizes of the groups are recorded which are m and n. Then, it combines the groups and assign them into pool. Run model part simulates the null hypothesis. I found my Pvalue by using these functions. My Pvalue was 0.937. This result is bigger than our threshold which is 0.05. That's why it is not statistically significant. So that we can reject the null hypothesis.



```
class HypothesisTest(object):

    def __init__(self, data):
        self.data = data
        self.MakeModel()
        self.actual = self.TestStatistic(data)

    def PValue(self, iters=1000):
        self.test_stats = [self.TestStatistic(self.RunModel())
                           for _ in range(iters)]

        count = sum(1 for x in self.test_stats if x >= self.actual)
        return count / iters # mean

    def TestStatistic(self, data):
        raise UnimplementedMethodException()

    def MakeModel(self):
        pass

    def RunModel(self):
        raise UnimplementedMethodException()
```



```
class DiffMeansPermute(thinkstats2.HypothesisTest):

    def TestStatistic(self, data):
        group1, group2 = data
        test_stat = abs(group1.std() - group2.std())
        return test_stat

    def MakeModel(self):
        group1, group2 = self.data
        self.n, self.m = len(group1), len(group2)
        self.pool = np.hstack((group1, group2))

    def RunModel(self):
        np.random.shuffle(self.pool)
        data = self.pool[:self.n], self.pool[self.n:]
        return data
```



```
data = t.won, t.lost
ht = DiffMeansPermute(data)
pvalue = ht.PValue()
pvalue
```

```
0.937
```

**11)** As a conclusion, in statistic there are some useful methods that can be used to see the relationship between variables. First, I made charts (histogram, PMF, CDF) to see my data. After that I visualized my data with time in scatter plot. It showed the relationship between variables. After that, I calculated correlation which was -0.815 which means the variables inversely correlated. My hypothesis was there is a relationship between the numbers of wins that team's get and their position in the league. My null hypothesis was there is no relationship between them. Then, to see if it happened by chance or not I made hypothesis testing and I found the Pvalue. What I understood from these calculations and from these results, if a team want to be in the higher position in the league they need to get more wins.