Cs 240 Project Report

Doğancan Gemici
214971336

Introduction

I start to check up on the Baseball Data because i don't have much idea about Baseball. So, i can learn new things with searching and reading about the game.Firstly, i read the what readme file to see what it includes and what can i work on it. For example, firstly i decided to work on this question : “What is the relationship between position and assists numbers? Do positions any have effects number of assists?”. So, i need to learn positions then i checked them on internet. However, i analyze many things like that. Then, my last decision question is **“is there any relationship between runs and hits? do number of runs effect number of hits?”.**

In my opinion “number of runs effect number of hits”. Lets have a look at that is true or not.

Firstly, i import the necessery libraries then import and open my data.

Then, i specified data which i am gonna work on it. I did this with .loc command. So, i can represent which columns i want.

```
In [53]: #this shows the columns that i am going to use.  I am gonna have a look relationship between position and assists average
myspecificdata = mydata.loc[:,['playerID', 'R', 'H']]
myspecificdata
```

8	barrebi01	1	1
9	barrofr01	13	13
10	bassjo01	18	27
11	battijo01	0	0
12	bealsto01	6	7
13	beaveed01	7	6
14	bechtge01	24	33
15	bellast01	26	32
16	berkena01	0	0
17	berryto01	0	1
18	berthha01	17	17
19	biermch01	0	0

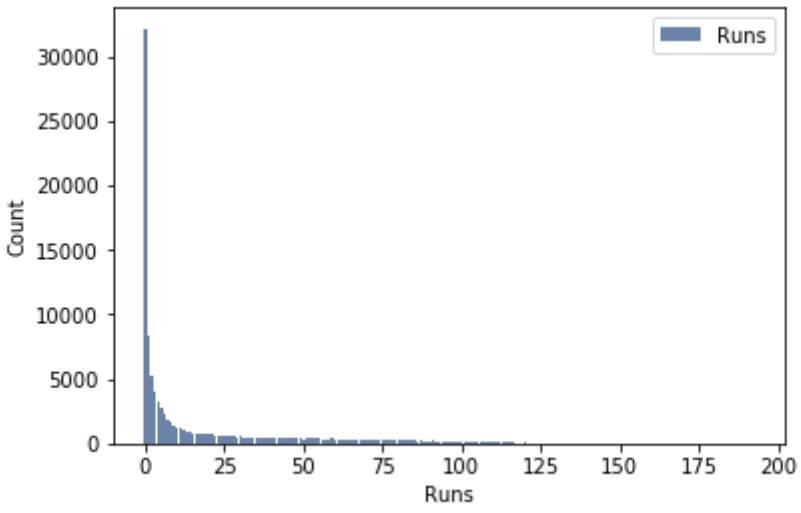
```
In [55]: #also .describe shows statistics about my data
myspecificdata.describe()
```

Out[55]:

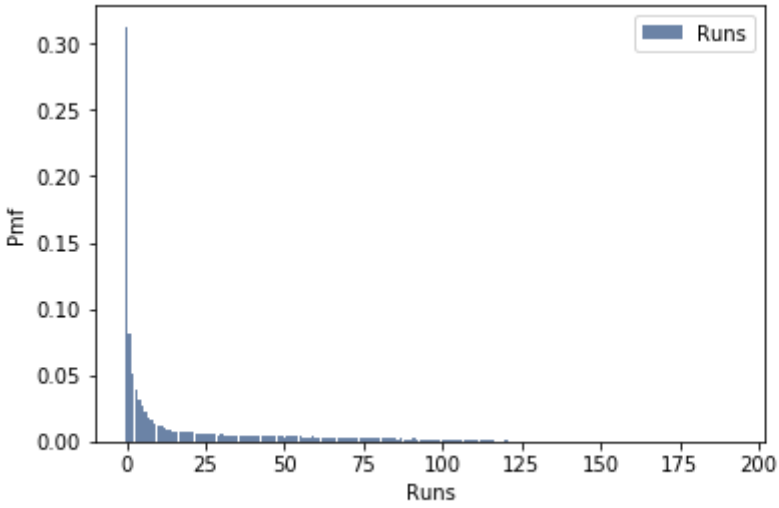
	R	H
count	102816.000000	102816.000000
mean	18.815544	37.139930
std	28.242983	52.603757
min	0.000000	0.000000
25%	0.000000	0.000000
50%	4.000000	9.000000
75%	27.000000	58.000000
max	192.000000	262.000000

```
In [56]: def Cov(xs, ys, meanx=None, meany=None):
    xs = np.asarray(xs)
```

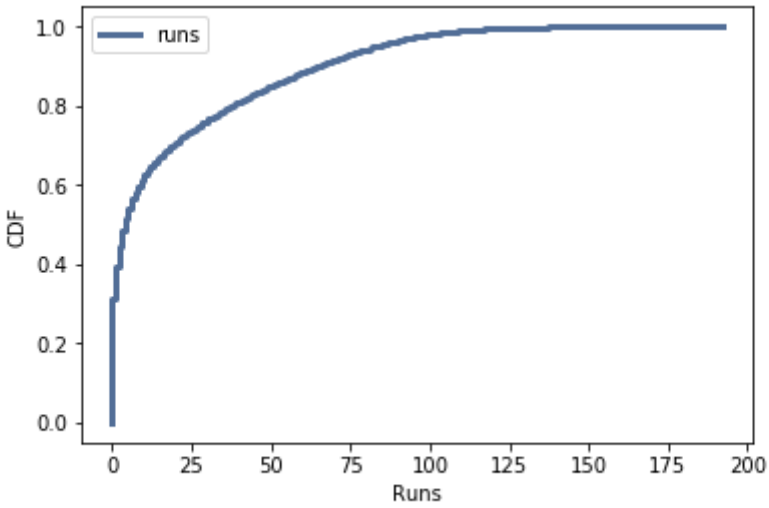
Then, i used describe command to see their statistics like mean, standard deviation(std) etc. To see how much similarity between them. After that, i represent 2 Histogram, 2 CDF, 2 PMF. Because i want to see their both functions. I will put one of them below but they are included in my codes file.



This is histogram for number of runs.

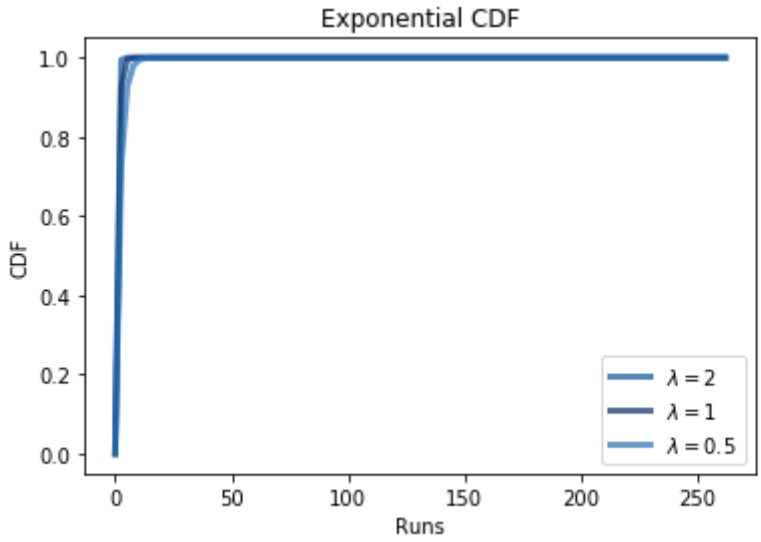


This bar charts show us probability mass function of runs.



This is cumulative distribution function of runs.

After i show these distributions i picked exponential distribution to model my data . Here's what the exponential CDF looks like with a range of parameters.



After i represent functions of my data, i calculated correlation between these two variables to observe how much they have relation or do they have relation. Correlation takes values between 1 and -1. If value get closer 1, it means there is positive correlation. If value of correlation get -1, it means there is still correlation but it is negative. For example, while the number of runs increase, if the number of hits also increase it means there is positive correlation. If the number of hits decrease while the number of runs increase it means there is negative correlation. Based on these informations i calculated correlatin and i got result like 0.96743369053648909 which means they have strongly positive correlation.

Then, i have last step to process make sure they have relationship.

I will do hypothesis testing. I have null hypothesis here like if number of runs increase, number of hits do not increase. And there is p-value.There is a p-value in hypothesis testing. I tried to find p value with 2 different ways to make sure i am in the true way.

In [71]:
```
class DiffMeansPermute(thinkstats2.HypothesisTest):

    def TestStatistic(self, data):
        myspecificdata.R, myspecificdata.H = data
        test_stat = abs(myspecificdata.R.std() - myspecificdata.H.std())
        return test_stat

    def MakeModel(self):
        myspecificdata.R, myspecificdata.H = self.data
        self.n, self.m = len(myspecificdata.R), len(myspecificdata.H)
        self.pool = np.hstack((myspecificdata.R, myspecificdata.H))

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

In [72]:
```
ht = DiffMeansPermute(data)
pvalue = ht.PValue()
pvalue
```

Out[72]: 0.0

In [24]:
```
class CorrelationPermute(thinkstats2.HypothesisTest):

    def TestStatistic(self, data):
        xs, ys = data
        test_stat = abs(thinkstats2.Corr(xs, ys))
        return test_stat

    def RunModel(self):
        xs, ys = self.data
        xs = np.random.permutation(xs)
        return xs, ys
```

In [25]:
```
data = myspecificdata.R, myspecificdata.H
correlationtest = CorrelationPermute(data)
pvalue = correlationtest.PValue()
pvalue
```

Out[25]: 0.0

I used difference means permute function but i replaced means with standard deviation in teststatistic part. Then, i got pvalue as a 0. Then i calculated p value again with different function which is correlation permutation. Then i got again 0. So, i made sure p value is zero.

Conclusion

As i explained steps above, i followed these steps and i got finally p value. The founded p-value is 0, which means that in 2000 trials i didn't see a correlation, under the null hypothesis, that exceeded the observed correlation. That means that the p-value is probably smaller than 1/10001/2000 , but it is not actually 0. So, our hypothesis seems true. It means there is strongly (as we get this information from correlation) positive relationship between these two variables. Number of runs effect the number of hits. So, if number of runs are many, then number of hits also many.