CS 240 Final Project Report

Mehmet Arif ERGÜN

Part 1

1. What is the relation between total time played and scores ?

2. How the conference of a player affects scores of the player in all stars?

The data shows us there is no significant difference between scores of players from east and west conferences at all stars.

```
playeras_data.groupby(["conference"])["points"].mean()
```

```
conference
Allstars    11.500000
Denver      13.090909
East        10.793238
West        10.661039
```

3. How total scored points affects the win rate of the team ?

To answer that question, I simply divide teams to two groups by their score, then calculate their means. I as can be seen below, teams that scores higher points have higher win averages.

```
lower_points = team_data[team_data["o_pts"] < 8321]
higher_points = team_data[team_data["o_pts"] >= 8321]
print lower_points["won"].mean()
print higher_points["won"].mean()
```

```
32.4268929504
42.6519480519
```

4. Is there a significant difference between total points scored by teams from different divisions ?

As can be seen from the list below, if we ignore the differences caused by the divisions that are no longer exist, there more than 700 points difference between minumum and maximum averages.

```
team_data.groupby(["divID"])["o_pts"].mean()
```

```
divID
AT    8214.824034
CD    8162.931559
EA    2597.851064
ED    7878.031250
MW    8426.731343
NO     364.875000
NW    7958.450000
PC    8499.500000
SE    7785.850000
SO     339.750000
SW    7831.575000
WD    7881.212500
WE    2482.760870
```

I have chosen the question 1 to analyze.

Hypothesis: Players who plays for longer durations scores higher points.

Part 2

To test my hypothesis I will use basketball_players data set from provided ones. Columns that will be used are: PlayerID, minutes, points from that data frame. Image below shows how my dataframe looks.

```
hypothesis_data = player_data.filter(["playerID", "minutes","points"])
hypothesis_data.tail(10)
```

	playerID	minutes	points
23741	blyesy01	975	496
23742	chmiemo01	528	208
23743	clarkbo01	217	59
23744	hillcl01	422	145
23745	johnsan01	732	314
23746	kaisero01	978	467
23747	spragbr01	746	356
23748	tayloro02	1007	355
23749	wellsra01	36	4
23750	wrighle01	813	195

To clean the data I dropped the rows that has NaN values for the columns that I interested. Then I cleaned the rows that has 0 in ‘minutes’ column because they are mostly unknown data.

Part 3

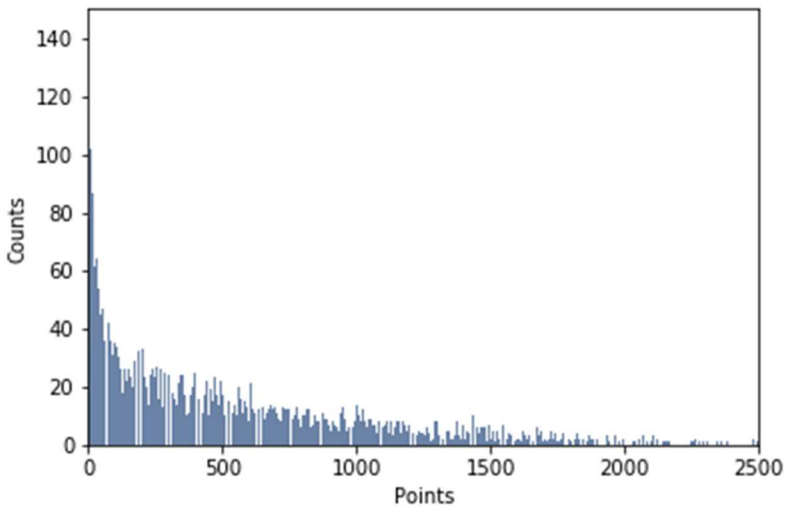
The descriptive statistics that are chosen can be seen in the image below.

```
#Print the descriptive statistics
print "Mean of scores %d" % hypothesis_data.points.mean()
print "Median of scores:%d" % hypothesis_data.points.median()
print "Standard deviation of scores: %d" % hypothesis_data.points.std()
print "Maximum of scores: %d" % hypothesis_data.points.max()
print "Minumum of scores: %d" % hypothesis_data.points.min()
```

```
Mean of scores 534
Median of scores:386
Standard deviation of scores: 513
Maximum of scores: 4029
Minumum of scores: 0
```

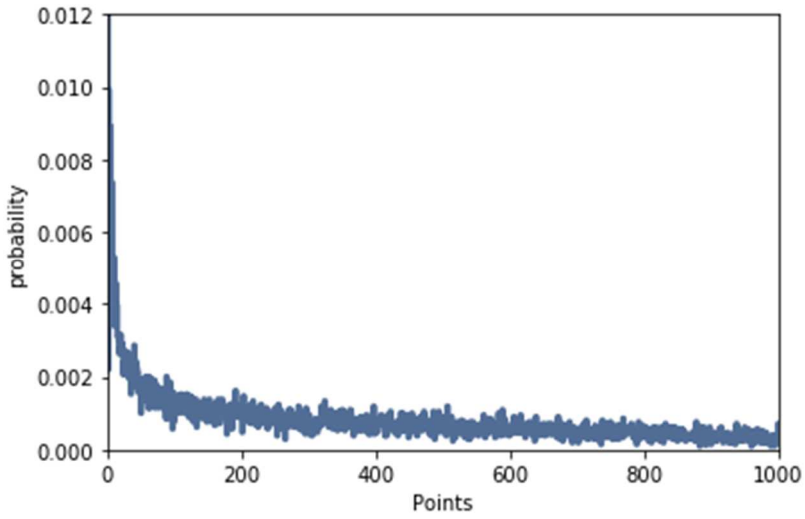
Histogram of the scores can be seen in figure below

```
#Create histogram and plot it using thinkplot module.
hist = thinkstats2.Hist(hypothesis_data.points)
thinkplot.Hist(hist)
thinkplot.Config(xlabel='Points',
ylabel='Counts',
axis = [0,2500,0,150])
```



Probability Mass Function

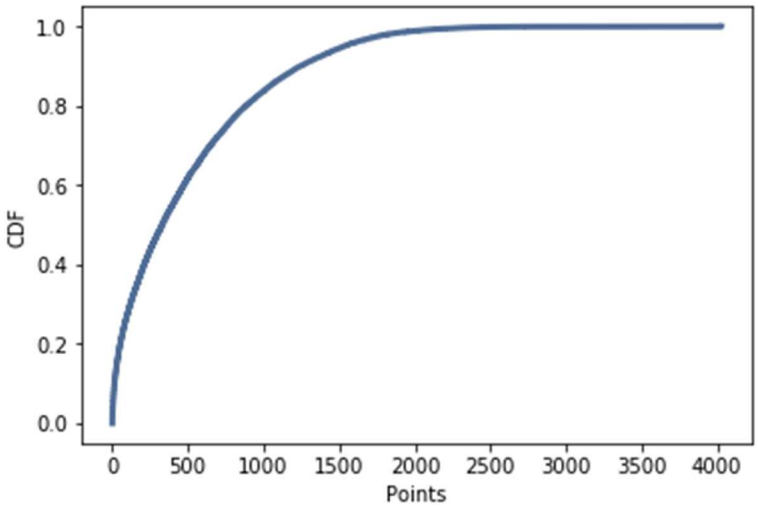
```
#Create pmf and plot it using thinkplot module.
pmf = thinkstats2.Pmf(hypothesis_data.points)
thinkplot.Pmf(pmf)
thinkplot.Config(xlabel='Points',
ylabel='probability',
axis = [0,1000,0,0.012])
```



Cumulative Distribution Function

```
#Create cdf and plot it using thinkplot molude.
cdf = thinkstats2.Cdf(hypothesis_data.points)
thinkplot.Cdf(cdf)

thinkplot.Config(xlabel = 'Points',ylabel = 'CDF')
```



Eventhough the histogram and pmf are nested because of the high variation of scores, it can be seen that the frequency is decreasing as the scores get higher and higher. CDF of 1000 is around 0.83 meaning that 83% of the scores are below 1000.

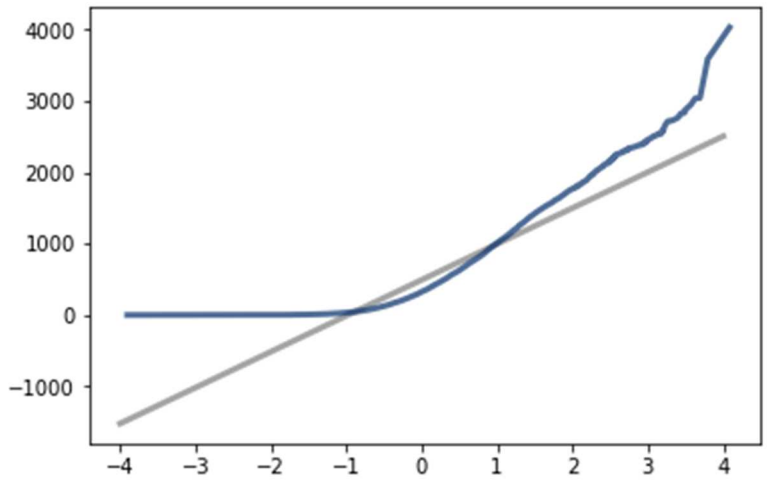
Part 4

To model the distribution of the scores, I used normal distribution. The figure below shows that how the distribution of models fits the n

```
#Function is taken from thinkstats to model the distribution of points.
def MakeNormalPlot(column):
    mean = column.mean()
    std = column.std()
    xs = [-4, 4]
    fxs, fys = thinkstats2.FitLine(xs, inter=mean, slope=std)
    thinkplot.Plot(fxs, fys, color='gray', label='model')
    xs, ys = thinkstats2.NormalProbability(column)
    thinkplot.Plot(xs, ys)

#Create model of distribution of models
MakeNormalPlot(hypothesis_data.points)
```

ormal distribution model.

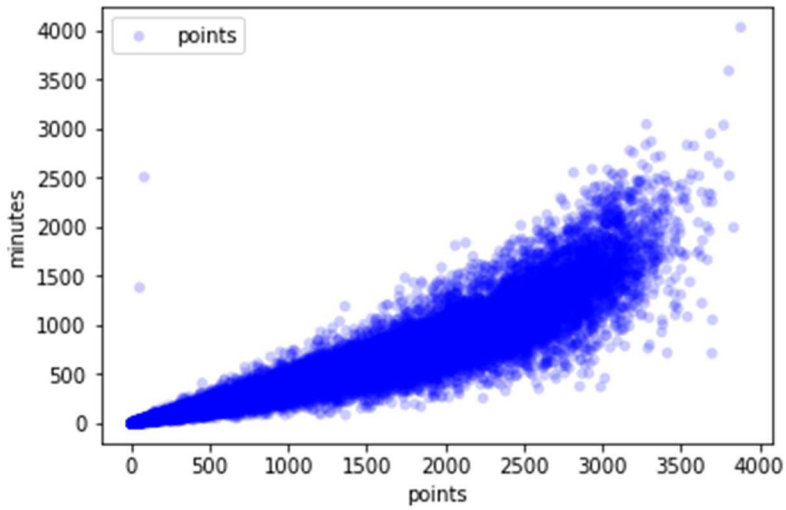


The distribution of the points fits the normal model mostly between [0,4]. The deviation in negative part is not demonstrative because scores can not be negative. The scores higher than 3000 are higher than what the model expects.

Part 5

I have chosen the minutes and points column to test the correlation. ScatterPlot from thinkplots is used to visualize the correlation. The figure below shows the scatter plot.

```
#Scatter plot to visualize the correlation between minutes and scores.
thinkplot.Scatter(hypothesis_data.minutes, hypothesis_data.points)
thinkplot.Show(xlabel="points",
ylabel="minutes")
```




```
#Calculate the Pearson's correaltions using scipy module
correlation_coefficient,_ = scipy.stats.pearsonr(hypothesis_data.minutes, hypothesis_data.points)
print correlation_coefficient
```

```
0.92669007073
```

To quantify the correlation I used Pearson's Correlation from scipy. The correlation coefficient is 0.9266 supporting the results in scatter plot. There is a strong positive correlation between minutes and scores.

Part 6

```
#Craate a hypothesis test to test the correlation
class pointtest(thinkstats2.HypothesisTest):

    #Pearson's correlation is used as test statistic
    def TestStatistic(self,data):
        xy,xs = data
        test_stats = abs(scipy.stats.pearsonr(xy,xs)[0])
        return test_stats

    #Create random model by shuffling points
    def RunModel(self):
        xy,xs = self.data
        xs = np.random.permutation(xs)
        return xy,xs


data = hypothesis_data.minutes, hypothesis_data.points
ht = pointtest(data)
#Print the p-value and R squarred.
print ht.PValue()
print correlation_coefficient**2
```

```
0.0
0.85875448719
```

As stated in the first part of the report, my hypothesis is players who plays for longer durations scores higher points. To test my hypothesis I used HypothesisTest framework from thinkstats2.

The test statistic of the test is absolute value of pearson correlation coefficient between variables since I am testing the correlation. As random model, we shuffle the given data to simulate a world where the parameters are same as actual data but variables unrelated.

The p-value of our test is 0.0 meaning that the correlation between minutes and points is impossible to occur by chance. So my hypothesis is true.

Conclusion

In that project, the exploratory questions about basketball dataset are presented and short answers are given to three of them using simple statistics from the dataset. One of the questions is chosen for further analysis and inspections in terms of characteristic of the data. My hypothesis was players who plays for longer durations scores higher points. To test my hypothesis, firstly I inspected the statistical insigths of the points columns from data frame. The CDF of the column has shown that more than 80% of the players scored 1000 points or less. Then scatter plot is created to visualize the correlation between minutes and points columns. Pearson's correlation coefficient is 0.9266 showing the strong positive linear correlation between variables.

Lastly HypothesisTest framework of thinkstats2 was my tool to test my hypothesis. The 0.0 p-value shown that the correlation between mentioned variables cannot occur by chance. The coefficient of determination is 0.8587 indicating that variance in minutes can explain 85% of the variation of points. I concluded my project that my hypothesis was correct.