playerAnalysis_proj_CS105

March 19, 2020

Viraj Dhillon 862015754

For this phase of the project, I want to explore some analysis based on players stats. The first is that I want to be able to see what player statisitcs will increase a players minutes per game. I believe that a players playing time is mainly tied to the amount of points they average, and rebounds being a lower indicator of their playtime, and turnovers being the lowest indicator. We can use linear regression to see what the coefficients will be, and can see whether they are positive or negative.

Here we will be using the dataset that we've been using for the past two phases, "NBA1950-2019.csv". We will only be using players from 1980 and beyond because this is when the modern NBA started and statistics were more closely recorded.

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.model_selection import train_test_split
statsDF = pd.read_csv("NBA1950-2019.csv")
statsDF = statsDF.drop(columns = ["Unnamed: 0", "Unnamed: 0.1"])
statsDF = statsDF[(statsDF["Season"] > 1981)]
statsDF = statsDF.dropna(subset=['Player'])
statsDF = statsDF.fillna(0)
statsDF.head()
```

[1]:
```
                Player Pos   Age   Tm     G    GS    MP   FG   FGA    FG%  \
287      Alaa Abdelnaby  PF  26.0  TOT  54.0   0.0   9.4  2.2   4.3  0.511
288      Alaa Abdelnaby  PF  26.0  SAC  51.0   0.0   9.3  2.3   4.3  0.532
289      Alaa Abdelnaby  PF  26.0  PHI   3.0   0.0  10.0  0.3   3.7  0.091
290  Mahmoud Abdul-Rauf  PG  25.0  DEN  73.0  43.0  28.5  6.5  13.8  0.470
291       Michael Adams  PG  32.0  CHH  29.0   0.0  15.3  2.3   5.1  0.453

     …  ORB  DRB  TRB  AST  STL  BLK  TOV   PF   PTS  Season
287  …  0.7  1.4  2.1  0.2  0.3  0.2  0.8  1.9   4.7    1995
288  …  0.7  1.4  2.1  0.3  0.3  0.2  0.8  2.0   5.0    1995
289  …  1.0  1.7  2.7  0.0  0.0  0.0  1.7  0.7   0.7    1995
290  …  0.4  1.4  1.9  3.6  1.1  0.1  1.6  1.7  16.0    1995
291  …  0.2  0.8  1.0  3.3  0.8  0.0  0.9  1.4   6.5    1995

[5 rows x 30 columns]
```

Here we will be splitting up our data into testing and training. The predicted value we want to find is the average career amount of minutes a player plays in a game. The features we will be using is a players career average points, rebounds, assists, etc.

[2]:
```
y = statsDF.groupby('Player')['MP'].mean()
x = {'PTS': statsDF.groupby(["Player"])["PTS"].mean(),
     'STL': statsDF.groupby(["Player"])["STL"].mean(),
     'BLK':  statsDF.groupby(["Player"])["BLK"].mean(),
     'TRB': statsDF.groupby(["Player"])["TRB"].mean(),
     'TOV': statsDF.groupby(["Player"])["TOV"].mean(),
     'AST': statsDF.groupby(["Player"])["AST"].mean(),
     'GS': statsDF.groupby(['Player'])['GS'].mean().round()
    }
x = pd.DataFrame(data = x)
xTrain, xTest, yTrain, yTest = train_test_split(x, y, test_size = 0.3)
x.head()
```

[2]:
```
                    PTS       STL       BLK       TRB       TOV       AST  \
Player
A.C. Green     9.233333  0.805556  0.394444  7.333333  1.077778  1.050000
A.J. Bramlett  1.000000  0.100000  0.000000  2.800000  0.400000  0.000000
A.J. English   9.850000  0.400000  0.150000  2.100000  1.350000  2.150000
A.J. Guyton    3.800000  0.333333  0.133333  0.700000  0.666667  1.566667
A.J. Hammons   2.200000  0.000000  0.600000  1.600000  0.500000  0.200000

                 GS
Player
A.C. Green     50.0
A.J. Bramlett   0.0
A.J. English    9.0
A.J. Guyton     5.0
A.J. Hammons    0.0
```

Now let's start analyzing NBA stats and correlation to minutes played.

[6]:
```
from sklearn.linear_model import LinearRegression
model = LinearRegression()

xTrain = xTrain[["PTS", "AST", "TRB", "STL", "BLK", "TOV", "GS"]]
xTest = xTest[["PTS", "AST", "TRB", "STL", "BLK", "TOV", "GS"]]
model.fit(X = xTrain, y = yTrain)
yPredict = model.predict(X = xTest)
model.coef_
```

[6]:
```
array([ 0.78664048,  0.80003455,  0.94734759,  3.88720626, -0.28042806,
       -0.05229967,  0.07480931])
```

To my surprise, my hypothesis wasn't fully correct. I stated that a players career points average will be the highest coefficent, and rebounds will be a lower coefficient, and turnovers will be the lowest. But in reality, it turns out that steals is the highest coefficient, and then followed by total rebounds. This makes sense since steals give teams momentum and coaches won't take players out after they commit steals. The points category is actually the third highest coefficient. Not surprisingly, turnovers is the lowest coefficient as players who have higher turnovers will get less playing time.

First: Steals

Second: Total Rebounds

Third: Assists

For the next part of this phase, I want to estimate how much an NBA player should be paid. To do this, I imported another dataset that has NBA salaries from 1990-2017. The dataset can be found at: https://www.kaggle.com/whitefero/nba-player-salary-19902017.

In the following code, I clean up the salary column, and then insert the salaries of a player to our stats dataframe.

[4]:
```
#clean up the salary column, getting rid of $, and commas
salaryDF = pd.read_csv("Player - Salaries per Year (1990 - 2017).csv")
salaryDF[" Salary in $ "] = salaryDF[" Salary in $ "].str.replace(',', '')
salaryDF[" Salary in $ "] = salaryDF[" Salary in $ "].str.replace('$', '')
salaryDF = salaryDF.iloc[:-1]
salaryDF[" Salary in $ "] = salaryDF[" Salary in $ "].astype('float64')
statsDF = statsDF[(statsDF["Season"] > 1990) & (statsDF["Season"] <= 2018)]

statsDF = statsDF.drop_duplicates(subset=['Player', 'Season'])

statsDF["Salary"] = 0
#updating every players salary for a particular year
for index in statsDF.index:
    salaryPrice = salaryDF[(salaryDF['Player Name'] == statsDF.at[index,␣
 ↪'Player']) &
                                              (salaryDF['Season End'] == statsDF.
 ↪at[index, 'Season'])][' Salary in $ ']
    #if player played for multiple teams in one season, add the salaries up
    if(len(salaryPrice) > 0):
        salaryPrice = sum(salaryPrice)
    elif(len(salaryPrice) == 0): #salaries cannot be found
        continue
    statsDF.at[index, 'Salary'] = salaryPrice

statsDF = statsDF[statsDF['Salary'] != 0]#drop rows where salaries can't be␣
 ↪found
statsDF.head(10)
```

[4]:
```
                   Player Pos   Age   Tm     G    GS    MP   FG   FGA    FG%  \
287       Alaa Abdelnaby  PF  26.0  TOT  54.0   0.0   9.4  2.2   4.3  0.511
```

```
290  Mahmoud Abdul-Rauf  PG  25.0  DEN  73.0  43.0  28.5  6.5  13.8  0.470
291       Michael Adams  PG  32.0  CHH  29.0   0.0  15.3  2.3   5.1  0.453
292      Rafael Addison  SF  30.0  DET  79.0  16.0  22.5  3.5   7.4  0.476
293         Danny Ainge  SG  35.0  PHO  74.0   1.0  18.6  2.6   5.7  0.460
294    Victor Alexander   C  25.0  GSW  50.0  29.0  24.7  4.6   8.9  0.515
295      Derrick Alston   C  22.0  PHI  64.0   1.0  16.1  1.9   4.0  0.465
296       Greg Anderson  PF  30.0  ATL  51.0   0.0  12.2  1.1   2.0  0.548
299     Willie Anderson  SG  28.0  SAS  38.0  11.0  14.6  2.0   4.3  0.469
300        Greg Anthony  PG  27.0  NYK  61.0   2.0  15.5  2.1   4.8  0.437

     …  DRB  TRB  AST  STL  BLK  TOV   PF   PTS  Season   Salary
287  …  1.4  2.1  0.2  0.3  0.2  0.8  1.9   4.7    1995   650000
290  …  1.4  1.9  3.6  1.1  0.1  1.6  1.7  16.0    1995  2200000
291  …  0.8  1.0  3.3  0.8  0.0  0.9  1.4   6.5    1995  1240000
292  …  2.2  3.1  1.4  0.7  0.3  1.0  3.0   8.3    1995   250000
293  …  1.1  1.5  2.8  0.6  0.1  1.1  2.1   7.7    1995  2080000
294  …  4.1  5.8  1.2  0.6  0.6  1.5  2.9  10.0    1995  1377500
295  …  1.9  3.4  0.5  0.6  0.5  0.8  1.7   4.7    1995   150000
296  …  2.5  3.7  0.3  0.5  0.6  0.6  2.0   2.9    1995   510000
299  …  1.1  1.4  1.4  0.7  0.3  1.0  1.9   4.9    1995  2075000
300  …  0.9  1.0  2.6  0.8  0.1  0.9  1.6   6.1    1995  1471000

[10 rows x 31 columns]
```

Here is where I will be doing the linear regression of a players salary. I will be taking in various stats of a player and predict what the players pay should be.

[5]:
```python
salaryModel = LinearRegression()

salaryModel.fit(statsDF[['PTS', 'AST','TRB','STL','BLK', "TOV", 'MP', 'GS']], statsDF['Salary'])
print("Players predicted salary is: ", salaryModel.predict([[24, 6, 3, 1, .6, 0, 1, 82]])) #inputting a players stats
coef = salaryModel.coef_
stats = ['PTS', 'AST', 'TRB','STL','BLK', "TOV", 'MP', 'GS']
print("\nCoefficients")
for index in range(0, len(coef)):
    print(stats[index], ':', coef[index])
```

```
Players predicted salary is:  [16669233.70274856]

Coefficients
PTS : 451834.0049994652
AST : 731683.2001195339
TRB : 528671.1986827849
STL : -1346557.0371847418
BLK : 325382.32720202423
TOV : -1300228.0019259164
```

```
MP : -92692.26300974889
GS : 5416.320302613778
```

Above is the linear regression model that can predict a players salary based on the players stats. I chose the core stats of an NBA player, points, assists, rebounds, steals, blocks, turnovers, minutes played, and games started. Surprisingly, some features give negative coefficients when determining a players salary. One of these features is steals, with a coefficient of -1346557.0371847418. I would figure steals would help a players salary, as it is a positive indication of a players salary in real life.

Some outliers in data could be attributed to this. Michael Jordan is considered one of the best of all time, and in 1992 he was leading his team to playoffs and even won Most Valuable Player award. His stats were amazing, but he only got a 4.5 million dollar contract. Players like Jordan could be under bad or rookie contracts and have good numbers. Inflation and the raise of a salary cap also play a part in hand because NBA players started to get higher salaries. Bench players in today's NBA are getting higher salaries than Jordan did in 1992.

Regardless, this model does a good job of giving a good base estimate of how much a player should be paid.

[ ]:

```

```