

Flask Deployment

Predicting NBA salaries based on statistics

Submitted by Anthony Sanogo 28 January 2023

Executive Summary

OBJECTIVE: Provide actionable insights to help NBA players ask for deserving salary based on their stats.





Agenda

- Data Exploration
- Model Creation
- Writing Flask Application
- HTML Template Creation & Styling
- Commit to GitHub Repo
- Heroku Deployment



Data Exploration



Data Exploration

Below is the list of datasets used for the project:

- hba_2020_stats.csv this file contains every NBA player per game stat line for the 2019-2020 season
- ➤ nba_2020_all_stars.csv this file includes the names of the players that were deemed an All-Star for the 2019-2020 season
- ➤ nba-salaries.csv this file contains every NBA player per game salary for the 2019-2020 season

Stats file

I used web scraping to get every player's stat line for the 2020 season.

NBA Stats Dataset

Web Scraping from Basketball Reference

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager

driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))
driver.get("https://www.basketball-reference.com/leagues/NBA_2020_per_game.html")

html = driver.page_source

tables = pd.read_html(html)
df = tables[18]

driver.close()
```

Stats file (continue)

Cleaning and Optimizing the data

```
#Delete the repeated headers
#Only keep the total season stat line if the player was traded
df=df.drop duplicates()
df=df.drop duplicates(subset=['Player'], keep='first')
df=df.loc[~df['GS'].str.match("GS")]
#Drop unnecessary colums and reset the index
df=df[['Player','FG%','3P%','FT%','PTS','AST','TRB','BLK','STL']]
#Replace NaN Values with Zero
df=df.fillna(0)
#Change columns data type
df = df.astype({"Player": str,"FG%": float,"3P%": float,"FT%": float,"TRB": float,"AST": float,
               "STL": float, "BLK": float, "PTS": float})
#changing fields decimal to percentage
df['FG%'] = df['FG%']*100
df['3P\%'] = df['3P\%']*100
df['FT%'] = df['FT%']*100
df.to csv(r'nba 2020 stats.csv',index=False, encoding='utf-8', header='true')
df stats=df
df stats
```

All Star file

I used web scraping to get the name of the players selected to be an All-Star for the 2020 season.

NBA All-Star List Dataset

Web Scraping from Basketball Reference

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager

driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))
driver.get("https://www.basketball-reference.com/allstar/NBA_2020.html")

html = driver.page_source

tables1 = pd.read_html(html)
driver.close()
```

All Star file (continue)

Cleaning and Optimizing the data

```
#Put east and west player roster together
result = pd.concat([tables1[19], tables1[20]], axis=1, join="inner")
result = result.droplevel(0,axis=1)
#Keep players column
result = result[['Starters']]
#Remove unecessary rows
result = result[~result['Starters'].isin(["Reserves", "Team Totals"])]
#Drop null rows
result = result.dropna()
#Remove duplicate headers
result.columns = result.iloc[0]
result = result[1:]
#Add headers
result = result.T.reset index().T.reset index(drop=True)
result.columns = ["West", "East"]
result.to csv(r'nba 2020 all stars.csv',index=False, encoding='utf-8', header='true')
#Create nba 2020 all-star list
east = result['East'].values.tolist()
west = result['West'].values.tolist()
players=east+west
```

Salaries

The NBA salaries dataset was available on GitHub

NBA Salaries dataset

```
: #import salaries data
df_salaries = pd.read_csv('nba-salaries.csv')

df_salaries = df_salaries.loc[df_salaries['season'] == 2020]

#Drop unnecessary colums and reset the index
df_salaries=df_salaries.drop(['rank','position','team','season'], axis=1)

#Change columns data type
df_salaries = df_salaries.astype({"name": str,"salary": int})
df_salaries.to_csv(r'nba_2020_salaries.csv',index=False, encoding='utf-8', header='true')

#Rename unnamed columns
df_salaries.rename(columns={'name': 'Player', 'salary': 'Salary'}, inplace=True)
df_salaries
```

Final data after merge

	Player	FG%	3P%	FT%	PTS	AST	TRB	BLK	STL	All-Star	Salary
141	Aaron Gordon	43.7	30.8	67.4	14.4	3.7	7.7	0.6	8.0	0	19863636
174	Aaron Holiday	41.4	39.4	85.1	9.5	3.4	2.4	0.2	8.0	0	2329200
283	Abdel Nader	46.8	37.5	77.3	6.3	0.7	1.8	0.4	0.4	0	1618520
347	Admiral Schofield	38.0	31.1	66.7	3.0	0.5	1.4	0.1	0.2	0	1000000
180	Al Horford	45.0	35.0	76.3	11.9	4.0	6.8	0.9	8.0	0	28000000
122	Yogi Ferrell	42.0	30.4	85.7	4.4	1.4	1.0	0.1	0.4	0	3150000
84	Zach Collins	47.1	36.8	75.0	7.0	1.5	6.3	0.5	0.5	0	4240200
228	Zach LaVine	45.0	38.0	80.2	25.5	4.2	4.8	0.5	1.5	0	19500000
360	Zhaire Smith	27.3	0.0	50.0	1.1	0.3	0.3	0.0	0.4	0	3058800
406	Zion Williamson	58.3	42.9	64.0	22.5	2.1	6.3	0.4	0.7	0	9757440

Model Creation



Model Creation

```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import pickle
from sklearn.preprocessing import StandardScaler
df = pd.read_csv('nba_stats&salaries&all_star_status.csv')
X = df.drop(['Salary', 'Player'], axis=1)
Y = df.Salary
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2)
X_train.shape, Y_train.shape
X_test.shape, Y_test.shape
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score
model = linear_model.LinearRegression()
model.fit(X_train, Y_train)
Y pred = model.predict(X test)
print('Coefficients:', model.coef_)
print('Intercept:', model.intercept_)
print('Mean squared error (MSE): %.2f'
      % mean_squared_error(Y_test, Y_pred))
print('Coefficient of determination (R^2): %.2f'
     % r2_score(Y_test, Y_pred))
pickle.dump(model, open('model.pkl','wb'))
```

Writing Flask Application



Writing Flask Application

```
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle
app = Flask( name )
model = pickle.load(open('model.pkl', 'rb'))
@app.route('/')
def home():
   return render template('index.html')
@app.route('/predict',methods=['POST'])
def predict():
   For rendering results on HTML GUI
   int features = [float(x) for x in request.form.values()]
   final features = [np.array(int features)]
   prediction = model.predict(final features)
   output = round(prediction[0], 2)
   return render template('index.html', prediction text='NBA Salary should be $ {}'.format(output))
if name == " main ":
   app.run(debug=True)
```

HTML Template Creation & Styling



HTML Template Creation & Styling

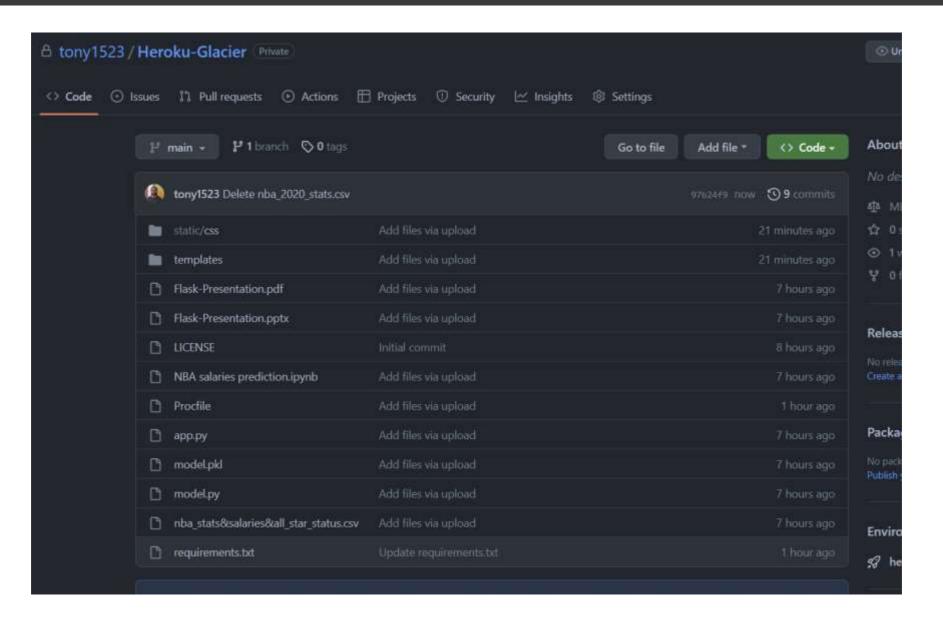
```
chtml>
       clink rel="stylesheet" href="../static/css/nba-salaries.css">
<body>
<h2> NBA Salary based on stats</h2>
cdiv class="maindiv">
  cform action="{{ url_for('predict')}}'method="post">
   <label for="Lname">FG%</label>
   cinput type="text" id="fg" name="FG%" pattern="^\d*(\.\d[0,2])?$"
   oninvalid="setCustomValidity('Please enter numbers')" placeholder="Enter field goal percentage per game..." required>,
    clabel for="lnowe">3P%c/label>
    cinput type="text" id="3p" name="3P%" pattern="^\d*(\.\d{0.2})?$"
    oninvalid="setCustomValidity('Please enter numbers')" placeholder="Enter 3 point percentage per game.." required>
    <label for="Lname">FT%</label>
    cinput type="text" id="ft" name="FT%" pattern="^\d*(\.\d[0.2])?$"
    oninvalid="setCustomValidity('Please enter numbers')" placeholder="Enter free throw percentage per game.." required>
    <label for="fname">Points</label>
    cinput type="text" id="pts" name="PTS" pattern="^\d"(\.\d{0,2})?$"
    oninvalid="setCustomValidity('Please enter numbers')" placeholder="Enter points per game.." required>
    <label for="Lname">Assists</label>
    cinput type="text" id="ast" name="AST" pattern="^\d*(\.\d{0.2})?$"
    oninvalid="setCustomValidity('Please enter numbers')" placeholder="Enter assists per game.." required>
   <label for="Lname">Rebounds</label>
   cinput type="text" id="trb" name="TRB" pattern="^\d*(\.\d{0,2})?$"
    oninvalid="setCustomValidity('Please enter numbers')" placeholder="Enter rebounds per game...." required>
    <label for="Lname">Blocks</label>
    cinput type="text" id="blk" name="BLK" pattern="^\d"(\.\d{0,2})?$"
    oninvalid="setCustomValidity('Please enter numbers')" placeholder="Enter blocks per game.." required>
    <label for="Lname">Steals</label>
    cinput type="text" id="stl" name="STL" pattern="^\d*(\.\d{0,2})?$"
    oninvalid="setCustomValidity('Please enter numbers')" placeholder="Enter steals per game.." required>
     <label for="Lname">All Star Selection (1 for yes and 0 for no) </label>
     <input type="number" id="all-star" name="All-Star" min="0" max="1" required>
   <input type="submit" value="Submit">
{{ prediction_text }}
</div>
```

```
input[type=text], select {
 width: 100%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
 border: 1px solid #ccc;
 border-radius: 4px:
  box-sizing: border-box;
input[type=submit] {
 width: 100%;
  background-color: #3097c4;
  color: white;
  padding: 14px 20px;
  margin: 8px 0;
  border: none;
 border-radius: 4px;
  cursor: pointer;
input[type=submit]:hover {
  background-color: #1218d4;
 .maindiv {
 border-radius: 5px;
 background-color: #f2f2f2;
  padding: 20px;
  padding-bottom: 0;
  margin: 200px;
  margin-top: 0;
  margin-bottom: 10px;
h2{
 margin-top: 20px;
  text-align: center;
```

Commit to GitHub Repo

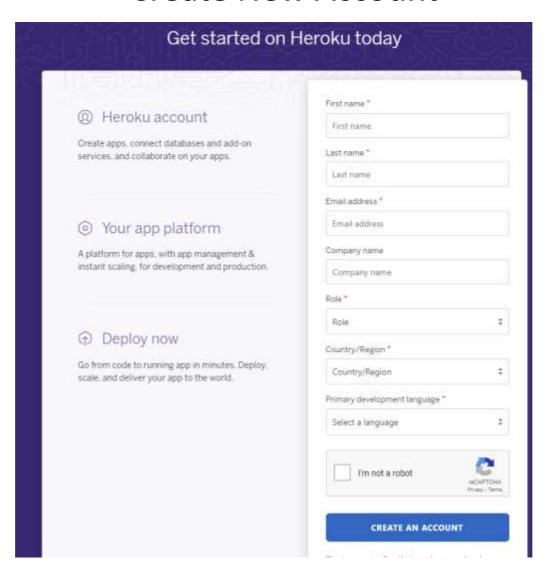


Commit to GitHub Repo

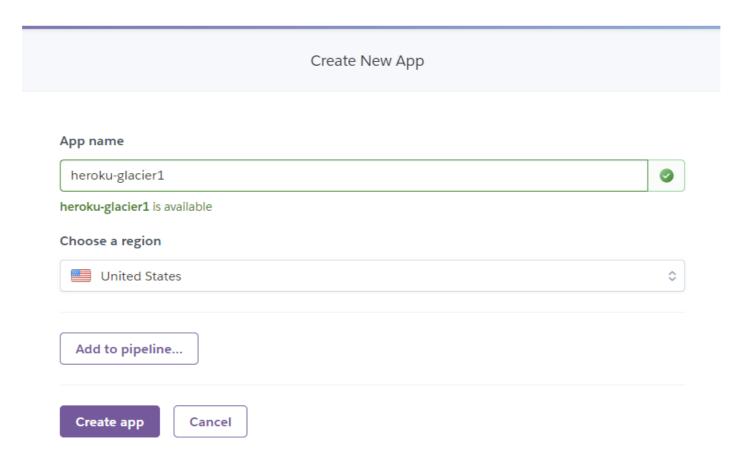




Create New Account



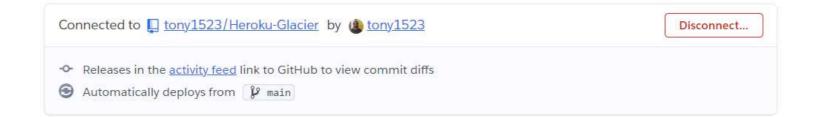
Create New App



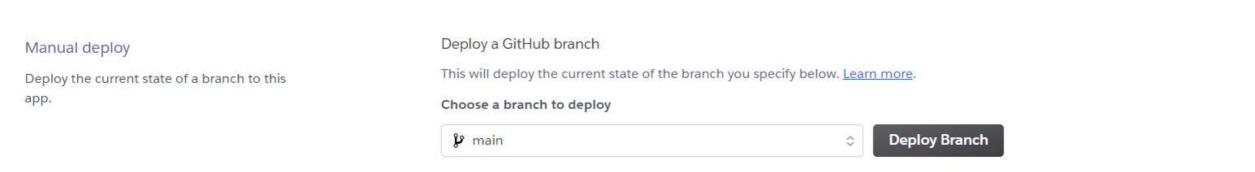
Connect app to repo



Code diffs, manual and auto deploys are available for this app.



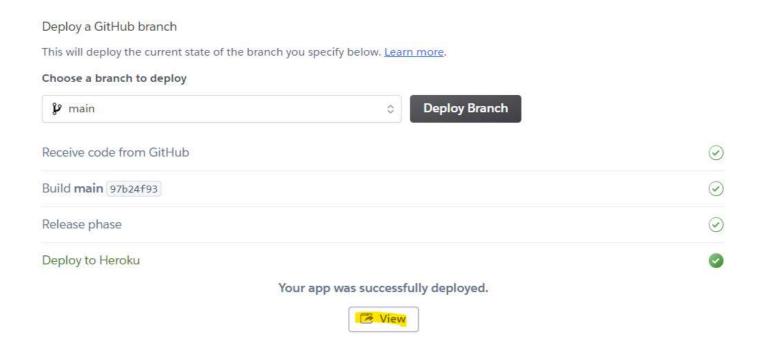
Deploy Branch

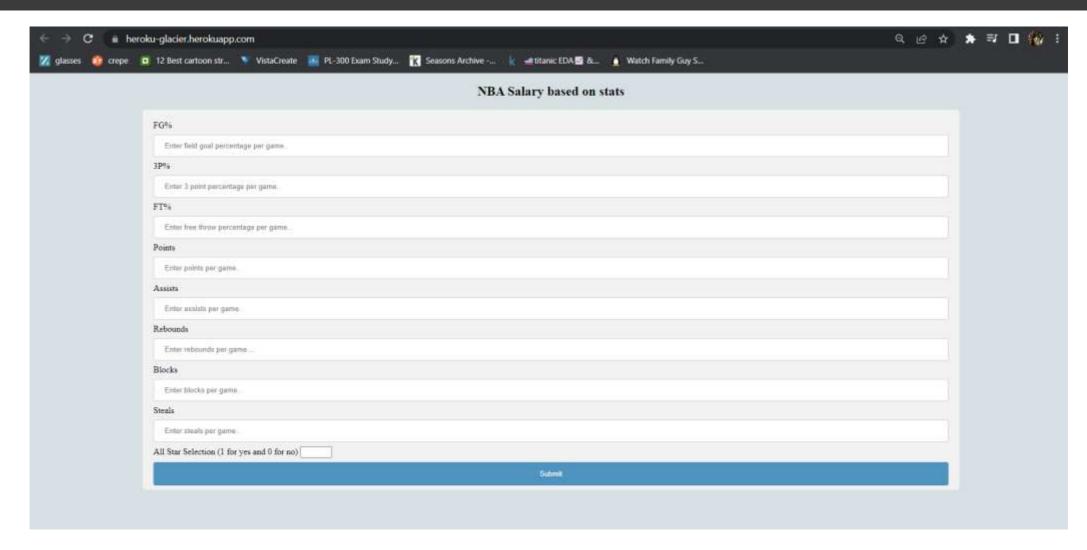


View App

Manual deploy

Deploy the current state of a branch to this app.





Web App Link: https://heroku-glacier.herokuapp.com/