Milestone 1: Data Collection, Preprocessing, and Exploratory Data Analysis (EDA) Report

By: Kuan-Chen, Chen

1. Topic selection

When exploring potential datasets for this project, I initially considered analyzing the relationships between Bitcoin, S&P 500, and gold prices. While these time-series datasets would have been straightforward to merge and analyze using traditional methods like ARIMA or Long-Short Term Memory (LSTM) models, I wanted to challenge myself with something novel and potentially more impactful.

The inspiration came while watching the Super Bowl with friends. During our discussion about how many talented high school athletes in the US gravitate toward American football over basketball, an intriguing question emerged: Could we use data science to help young athletes make more informed career choices?
This led to the conception of an athlete career prediction project. While both American football and basketball typically favor taller athletes, I expanded the scope to include soccer, where height isn't always deterministic of success. Take Lionel Messi, for instance – at 170cm (5'7"), he's become one of soccer's most celebrated players, demonstrating that peak performance isn't solely tied to physical stature.

The project culminates in an interactive dashboard with these key features:

- Input fields for user's physical metrics (height, weight, etc.)
- Salary prediction visualizations across different sports
- Career trajectory analysis showing potential earnings over time
- Sport-specific performance indicators
- Personalized recommendations based on physical attributes

The dashboard leverages machine learning models trained on professional athlete data from all three sports, providing users with data-driven insights about their

potential career paths. Through interactive visualizations, users can explore how different physical attributes correlate with success in each sport, and understand the financial implications of their career choices.

This approach not only helps young athletes make more informed decisions about their sporting careers but also provides interesting insights into how different sports value various physical attributes. The dashboard serves as a practical tool for career guidance while simultaneously revealing patterns in professional sports recruitment and compensation.

2. Data collection

Finding appropriate datasets proved to be more challenging than anticipated. While Kaggle offers numerous sports-related datasets, many were outdated or lacked crucial parameters for our analysis. This led to an innovative solution: leveraging data from sports video games.

Modern sports video games meticulously maintain accurate player statistics, physical attributes, and contract information, as they aim to provide realistic gameplay experiences. As an avid gamer myself and owner of NBA 2K25, I recognized that these games could serve as valuable data sources, since their databases are regularly updated to reflect real-world statistics and player developments.

1. **American Football**: Madden NFL 24 Player Ratings (https://www.kaggle.com/datasets/dtrade84/madden-24-player-ratings)
   Dimensions: 2,725 players × 71 attributes
   Key variables: Height, Weight, Speed, Strength, Position, Overall Rating

2. **Basketball**: Complete NBA 2K25 Player Dataset (https://www.kaggle.com/datasets/reinerjasin/nba-2k25-player-complete-dataset)
   Dimensions: 540 players × 53 attributes
   Key variables: Height, Weight, Salary, Position, Overall Rating, Potential

3. **Soccer**: FIFA Mobile FC-24
   (https://www.kaggle.com/datasets/rajatsurana979/fifafcmobile24)
   Dimensions: 17,000+ players × 42 attributes
   Key variables: Height, Weight, Position, Overall Rating, Market Value

However, the Madden NFL dataset lacked crucial salary information. To address this gap, I supplemented it with NFL contract data from Over The Cap, creating a contracts dataset. This supplementary dataset is available at:

- NFL Contracts: https://overthecap.com/contracts (https://drive.google.com/uc?export=download&id=1Rlz_djH1iBp6t4GbEPP8UJzE2-Kmh5nH)

All datasets are publicly accessible and used in compliance with their respective terms of use.

3. Data Preprocessing

After acquiring all necessary datasets using the Kaggle API and gdown library, we began the data preprocessing phase.

```
import kaggle
```

```
kaggle.api.authenticate()
```

```
kaggle.api.dataset_download_files('dtrade84/madden-24-player-ratings', path='.', unzip=True)
kaggle.api.dataset_download_files('reinerjasin/nba-2k25-player-complete-dataset', path='.', unzip=True)
kaggle.api.dataset_download_files('rajatsurana979/fifafcmobile24', path='.', unzip=True)
```

```
Dataset URL: https://www.kaggle.com/datasets/rajatsurana979/fifafcmobile24
Dataset URL: https://www.kaggle.com/datasets/dtrade84/madden-24-player-ratings
Dataset URL: https://www.kaggle.com/datasets/reinerjasin/nba-2k25-player-complete-dataset
```

```
url = 'https://drive.google.com/uc?export=download&id=1Rlz_djH1iBp6t4GbEPP8UJzE2-Kmh5nH'
gdown.download(url, 'nfl_salary.xlsx')
```

```
Downloading...
From: https://drive.google.com/uc?export=download&id=1Rlz_djH1iBp6t4GbEPP8UJzE2-Kmh5nH
To: /content/nfl_salary.xlsx
100%|██████████| 431k/431k [00:00<00:00, 6.60MB/s]
'nfl_salary.xlsx'
```

```
nfl_salary_raw = pd.read_excel('nfl_salary.xlsx')
nfl_raw = pd.read_csv('maddennf124fullplayerratings.csv')
nba_raw = pd.read_csv('current_nba_players.csv')
fifa_raw = pd.read_csv('male_players.csv')
```

A. American football

After loading our data into the program, we began cleaning and modifying the data according to our needs. Starting with American football, we first selected the essential features from the **Madden NFL 24 Player Ratings** dataset. I selected 'Full Name', 'Overall Rating', 'Height', 'Years Pro', 'Weight', 'Age', and 'Pos.' for our analysis. Next, we combined the **NFL Contracts** dataset with the **Madden NFL 24 Player Ratings** dataset by using an inner join on the players' names to merge them into a single dataset. Finally, we removed duplicates to ensure each player had only one entry in our dataset.

```
nfl = nfl_raw[['Full Name', 'Overall Rating', 'Height','Years Pro', 'Weight', 'Age','Pos.']]
nfl = nfl.merge(
        nfl_salary_raw,
        left_on='Full Name',
        right_on='Player',
        how='inner')

if not pd.api.types.is_numeric_dtype(nfl['APY']):
        nfl['APY'] = nfl['APY'].replace('[\$,]', '', regex=True).astype(float)
nfl.sort_values('APY', ascending=False, inplace=True)
nfl.drop_S('Full Name', keep='first', inplace=True)
nfl=nfl[['Full Name', 'Overall Rating', 'Height', 'Weight', 'Age','Years Pro', 'Pos.', 'APY']]
nfl
```

Later, we addressed the inconsistencies in height and weight measurements across the three datasets. Since each dataset used different measurement scales, we needed to standardize them for consistent analysis. I chose to convert all measurements to metric units: height in centimeters (cm) and weight in kilograms (kg). This required converting the initial measurements from inches and pounds to their metric equivalents.

```
def lbs_to_kg(weight_lbs):
        if pd.isna(weight_lbs):
                return None
        weight = pd.to_numeric(weight_lbs, errors='coerce')
        if pd.isna(weight):
                return None
        weight_kg = weight * 0.45359237
        return round(weight_kg)
```

```
def inches_to_cm(Height):
        if pd.isna(Height):
                return None
        Height = pd.to_numeric(Height, errors='coerce')
        if pd.isna(Height):
                return None
        height_cm = Height * 2.54
        return round(height_cm)
```

```
nfl['height_cm'] = nfl['Height'].apply(inches_to_cm)
nfl['weight_kg'] = nfl['Weight'].apply(lbs_to_kg)
nfl.drop(['Height', 'Weight'], axis=1, inplace=True)
nfl.rename(columns={'Full Name': 'name', 'Overall Rating': 'overall', 'Years Pro': 'years_pro', 'Pos.': 'position'}, inplace=True)
nfl= nfl[['name', 'overall', 'height_cm', 'weight_kg', 'Age', 'years_pro', 'position', 'APY']]
nfl
```

Finally, we standardized the column names and reordered them in a consistent way. This standardization made it easier to observe differences between sports and simplified the process of merging the three datasets later on.

	name	overall	height_cm	weight_kg	Age	years_pro	position	APY
752	Dak Prescott	87	188	108	30	7	QB	60000000.0
589	Joe Burrow	95	193	98	26	3	QB	55000000.0
1488	Jordan Love	70	193	99	24	3	QB	55000000.0
249	Trevor Lawrence	82	198	100	23	2	QB	55000000.0
16	Tua Tagovailoa	83	185	98	25	3	QB	53100000.0

B. Basketball

Next, we focused on processing the NBA 2K25 dataset. The primary challenge was that player ages weren't directly provided - instead, the raw data only contained 'birthdate'. We calculated current ages from this information, then converted the physical measurements from imperial units (feet/inches for height and pounds for weight) to our standardized metric units. Finally, we standardized the column names and order to maintain consistency across all datasets.

```
def feet_inches_to_cm(height_str):
    try:
        if pd.isna(height_str):
            return None
        height_str = str(height_str).replace(' ', '').replace('"', '')
        parts = height_str.split("'")
        feet = float(parts[0])
        inches = float(parts[1]) if len(parts) > 1 and parts[1] else 0
        total_inches = feet * 12 + inches
        cm = total_inches * 2.54
        return round(cm)
    except:S
        return None
```

```
target_date = pd.to_datetime('2024-08-30')
nba_raw['Age'] = pd.to_datetime(nba_raw['birthdate'], errors='coerce').apply(
    lambda x: relativedelta(target_date, x).years if pd.notnull(x) else None)
nba = nba_raw[['name', 'overall','height_feet', 'weight_lbs', 'Age','years_in_the_nba', 'position_1', 'season_salary']]
nba
```

```
nba['height_cm'] = nba['height_feet'].apply(feet_inches_to_cm)
nba['weight_kg'] = nba['weight_lbs'].apply(lbs_to_kg)
nba['APY']=nba['season_salary']
nba.drop(['height_feet', 'weight_lbs','season_salary'], axis=1, inplace=True)
nba.dropna(subset=['APY'], inplace=True)
nba.rename(columns={'Full Name': 'name', 'years_in_the_nba': 'years_pro', 'position_1': 'position'}, inplace=True)
nba= nba[['name', 'overall', 'height_cm', 'weight_kg', 'Age', 'years_pro', 'position', 'APY']]
nba
```

	name	overall	height_cm	weight_kg	Age	years_pro	position	APY
0	Aaron Gordon	85	203	107	28	11.0	PF	22841455.0
1	Aaron Holiday	74	183	84	27	7.0	PG	4668000.0
2	Aaron Nesmith	79	196	98	24	5.0	SF	11000000.0
3	Aaron Wiggins	79	196	86	25	4.0	PF	10514017.0
7	A.J. Green	77	193	86	24	3.0	SG	2120693.0

C. Soccer

Initially, we had to identify the appropriate version of the FIFA dataset to use. We selected FIFA24, as the number indicates the game's release year, meaning it contains 2024's player data. Next, we addressed the salary differences between soccer and the other two sports. Unlike NBA and NFL, soccer wages in the FIFA dataset ('wage_eur') are reported weekly, so we multiplied these values by 52 to calculate annual salaries.

We also encountered another consideration regarding league levels. While the NBA and NFL represent the highest and most prestigious level of competition in basketball and American football respectively, soccer has multiple competitive levels across different countries. To maintain comparable quality of competition across all three sports, we focused only on level 1 leagues from the FIFA dataset, which represent the top-tier professional competitions.

```
fifa = fifa_raw[fifa_raw['fifa_version'] == 24][['long_name', 'overall', 'height_cm','weight_kg', 'age', 'league_level','player_positions', 'wage_eur']]
fifa.drop_duplicates('long_name', keep='first', inplace=True)
fifa = fifa[fifa['league_level'] == 1]
fifa['Age'] = fifa['age']
fifa.drop('age', axis=1, inplace=True)
```

One limitation of the FIFA dataset is that it doesn't indicate the duration of a player's career in League 1. While the NBA and NFL datasets include 'Years Pro' data, this information is unavailable for soccer players, which could impact our ability to analyze career progression in the sport.

```
def wage_to_salary(wage_eur):
    return wage_eur*52
```

```
fifa['APY'] = fifa['wage_eur'].apply(wage_to_salary)
fifa.drop('wage_eur', axis=1, inplace=True)
fifa.rename(columns={'long_name': 'name', 'player_positions': 'position'}, inplace=True)
fifa= fifa[['name', 'overall', 'height_cm', 'weight_kg', 'Age', 'position', 'APY']]
fifa
```

	name	overall	height_cm	weight_kg	Age	position	APY
161671	Rúben Miguel Marques dos Santos Fernandes	71	187	81	37	CB	260000.0
161673	Cristiano Ronaldo dos Santos Aveiro	86	187	85	38	ST	3432000.0
161674	Karl Tommy Andreas Johansson	65	184	78	41	CB	44200.0
161675	José Manuel Reina Páez	77	188	92	40	GK	728000.0
161676	Roque Luis Santa Cruz Cantero	70	191	93	41	ST	26000.0

D. Merge

Finally, we going to merge these datasets into one datasets, because we already make the columns order same before, so we can merge these datasets very easily ,and we add column what sport it is at the

```
nfl['sport'] = 'NFL'
nba['sport'] = 'NBA'
fifa['sport'] = 'FIFA'
all_data = pd.concat([nfl, nba, fifa], ignore_index=True)
all_data
```

	name	overall	height_cm	weight_kg	Age	years_pro	position	APY	sport
0	Dak Prescott	87	188	108	30	7.0	QB	60000000.0	NFL
1	Joe Burrow	95	193	98	26	3.0	QB	55000000.0	NFL
2	Jordan Love	70	193	99	24	3.0	QB	55000000.0	NFL
3	Trevor Lawrence	82	198	100	23	2.0	QB	55000000.0	NFL
4	Tua Tagovailoa	83	185	98	25	3.0	QB	53100000.0	NFL
...	...	...	...	...	...	...	...	...	...
15555	Umut Yaşar Keçeci	58	182	73	19	NaN	ST	104000.0	FIFA
15556	Rakan Al Kaabi	54	174	71	20	NaN	CDM, CM	104000.0	FIFA
15557	Rodrigo Frutos	58	188	87	20	NaN	GK	26000.0	FIFA
15558	Christian Bos	55	180	70	18	NaN	RB	26000.0	FIFA
15559	Shahruddin Magomedaliyev	70	185	82	29	NaN	GK	364000.0	FIFA

4. Exploratory Data Analysis (EDA)

1. Descriptive statistics

For each sport in our dataset (NFL, NBA, and FIFA), we calculated descriptive statistics for the key physical and performance metrics: height (cm), weight (kg), age, overall rating, and annual pay (APY). This analysis provides a statistical overview of the player characteristics across these three major sports.

```
for sport in ['NFL', 'NBA', 'FIFA']:
    print(f"\nStats for {sport}:")
    sport_data = all_data[all_data['sport'] == sport]
    print(sport_data[['height_cm', 'weight_kg', 'Age', 'overall', 'APY']].describe().round(2))
```

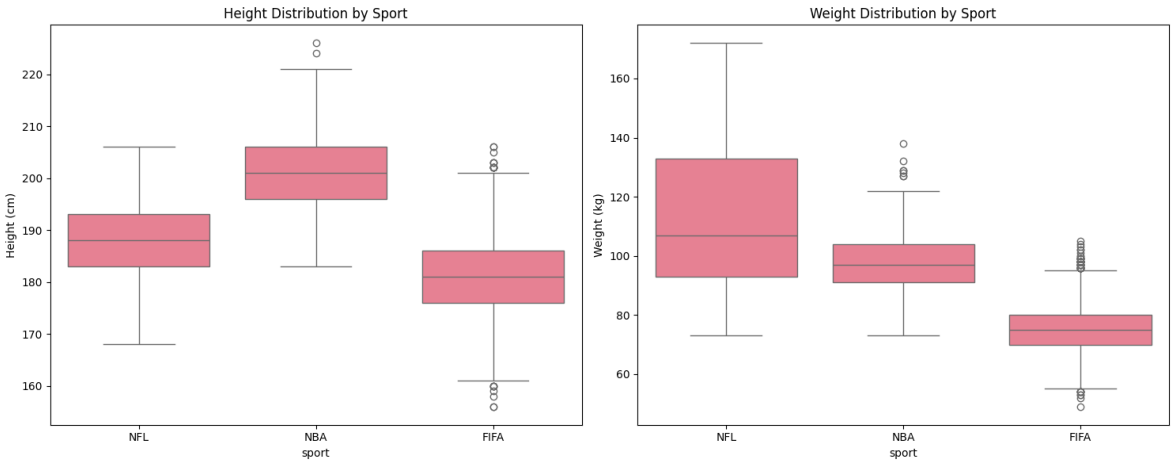
```
Stats for NFL:
       height_cm  weight_kg      Age  overall          APY
count    1629.00    1629.00  1629.00  1629.00      1629.00
mean      188.51     111.22    25.67    71.13   4580632.31
std         6.79      21.52     2.90    10.03   7309696.80
min       168.00      73.00    20.00    26.00         0.00
25%       183.00      93.00    24.00    65.00   1039115.00
50%       188.00     107.00    25.00    70.00   1345702.00
75%       193.00     133.00    27.00    77.00   4365448.00
max       206.00     172.00    39.00    99.00  60000000.00

Stats for NBA:
       height_cm  weight_kg      Age  overall          APY
count     399.00     399.00   399.00   399.00       399.00
mean      199.91      97.63    25.89    78.20  12590740.68
std         8.02      10.64     4.58     6.06  12843565.18
min       183.00      73.00    19.00    68.00    600000.00
25%       196.00      91.00    22.00    74.00   2995020.00
50%       201.00      97.00    25.00    77.00   7723000.00
75%       206.00     104.00    29.00    81.00  17130000.00
max       226.00     138.00    39.00    98.00  55761216.00

Stats for FIFA:
       height_cm  weight_kg       Age   overall          APY
count   13532.00   13532.00  13532.00  13532.00     13532.00
mean      181.42      75.06     25.42     66.33    528419.07
std         6.96       7.05      4.82      7.24   1104040.67
min       156.00      49.00     16.00     47.00     26000.00
25%       176.00      70.00     22.00     62.00     52000.00
50%       181.00      75.00     25.00     66.00    156000.00
75%       186.00      80.00     29.00     71.00    468000.00
max       206.00     105.00     43.00     91.00  18200000.00
```

2. Physical attributes

To visualize the physical differences between athletes across sports, we created box plots comparing height and weight distributions

```python
def plot_physical_attributes(df):
    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(15, 6))
    sns.boxplot(x='sport', y='height_cm', data=df, ax=ax1)
    ax1.set_title('Height Distribution by Sport')
    ax1.set_ylabel('Height (cm)')

    sns.boxplot(x='sport', y='weight_kg', data=df, ax=ax2)
    ax2.set_title('Weight Distribution by Sport')
    ax2.set_ylabel('Weight (kg)')

    plt.tight_layout()
    plt.show()
```

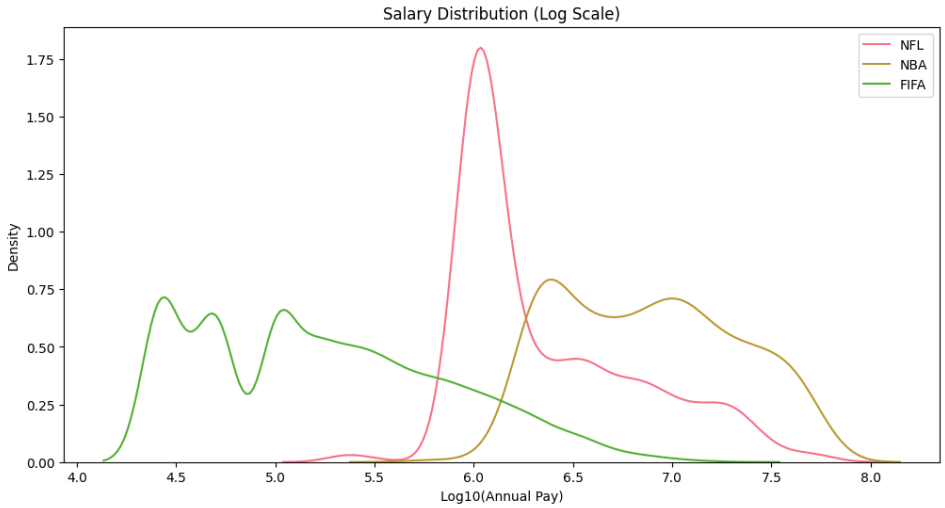
```python
plot_physical_attributes(all_data)
```



3. Salary distribution

We used a logarithmic scale for the salary data to better visualize the distribution, as player salaries can vary by orders of magnitude.

```python
def plot_salary_distribution(df):
    plt.figure(figsize=(12, 6))

    for sport in df['sport'].unique():
        sport_data = df[df['sport'] == sport]['APY']
        sns.kdeplot(data=np.log10(sport_data), label=sport)

    plt.title('Salary Distribution (Log Scale)')
    plt.xlabel('Log10(Annual Pay)')
    plt.ylabel('Density')
    plt.legend()
    plt.show()
```

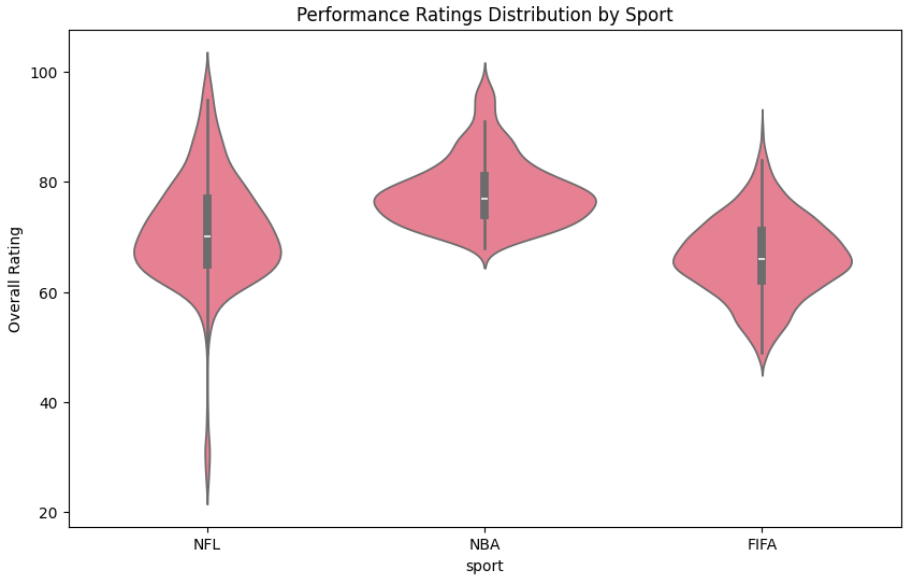


4. Plot performance comparison

This visualization uses violin plots to display the full distribution of player performance ratings in each sport. The width of each "violin" shows the density of players at each rating level, while the plot's shape reveals the complete distribution pattern. This helps us understand not only the median performance levels but also how player ratings are distributed within each sport, including any potential clusters or patterns in the data.

```
def  plot_performance_comparison(df):
        plt.figure(figsize=(10,  6))

        sns.violinplot(x='sport',  y='overall',  data=df)
        plt.title('Performance  Ratings  Distribution  by  Sport')
        plt.ylabel('Overall  Rating')

        plt.show()
```

```
plot_performance_comparison(all_data)
```



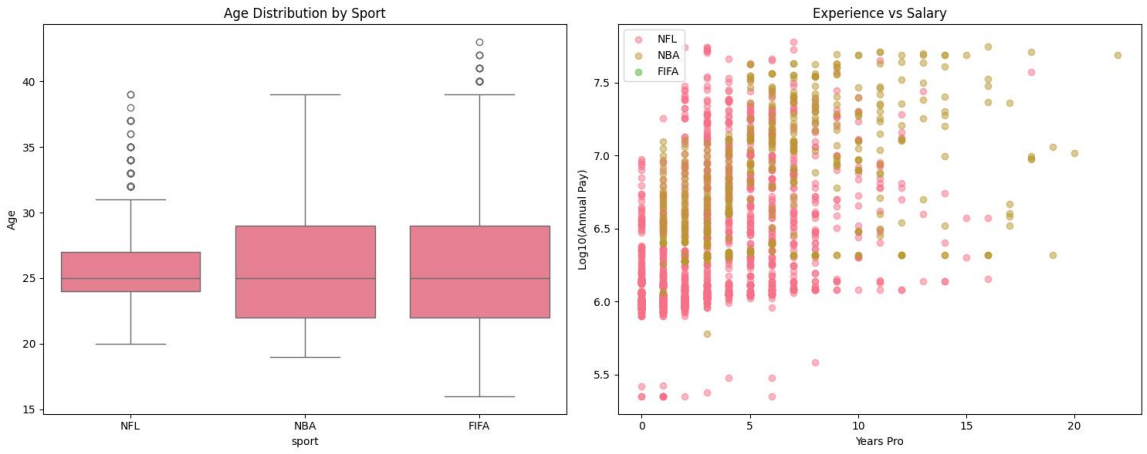
5. Age demographics and the relationship between experience and compensation across sports

The left panel uses a box plot to display the age distribution of players in each sport, showing the median age and quartile ranges. The right panel features a scatter plot that reveals how professional experience relates to salary levels, with different colors for each sport and a logarithmic scale for salary to better visualize the wide range of compensation levels. The transparency setting (alpha=0.5) helps visualize overlapping data points, making patterns more apparent in dense regions.

```
def plot_age_experience_analysis(df):
    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(15, 6))

    sns.boxplot(x='sport', y='Age', data=df, ax=ax1)
    ax1.set_title('Age Distribution by Sport')
    for sport in df['sport'].unique():
        sport_data = df[df['sport'] == sport]
        ax2.scatter(sport_data['years_pro'], np.log10(sport_data['APY']),
                    alpha=0.5, label=sport)

    ax2.set_title('Experience vs Salary')
    ax2.set_xlabel('Years Pro')
    ax2.set_ylabel('Log10(Annual Pay)')
    ax2.legend()
    plt.tight_layout()
    plt.show()
```

```
[ ] plot_age_experience_analysis(all_data)
```

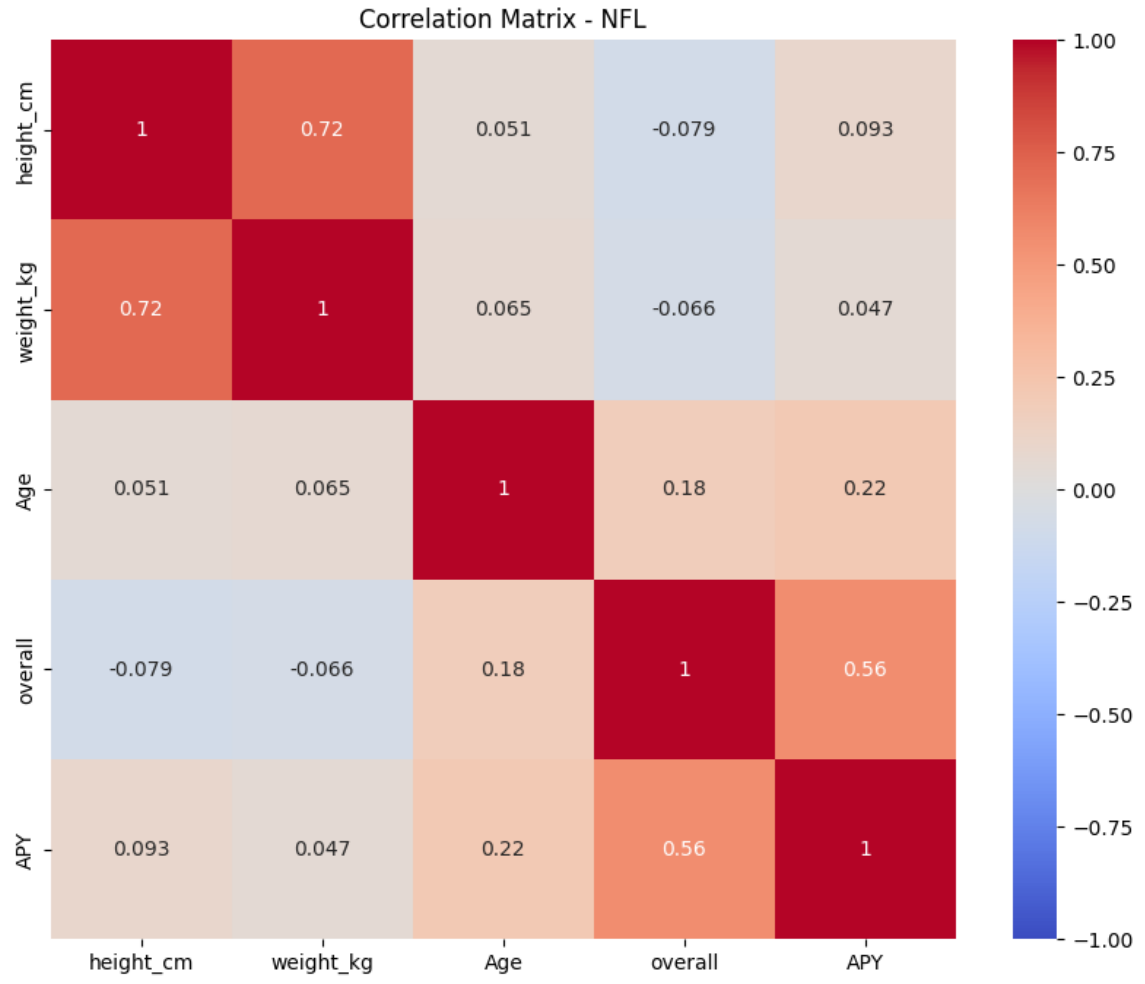
```
/usr/local/lib/python3.11/dist-packages/pandas/core/arraylike.py:399: RuntimeWarning:
divide by zero encountered in log10
```

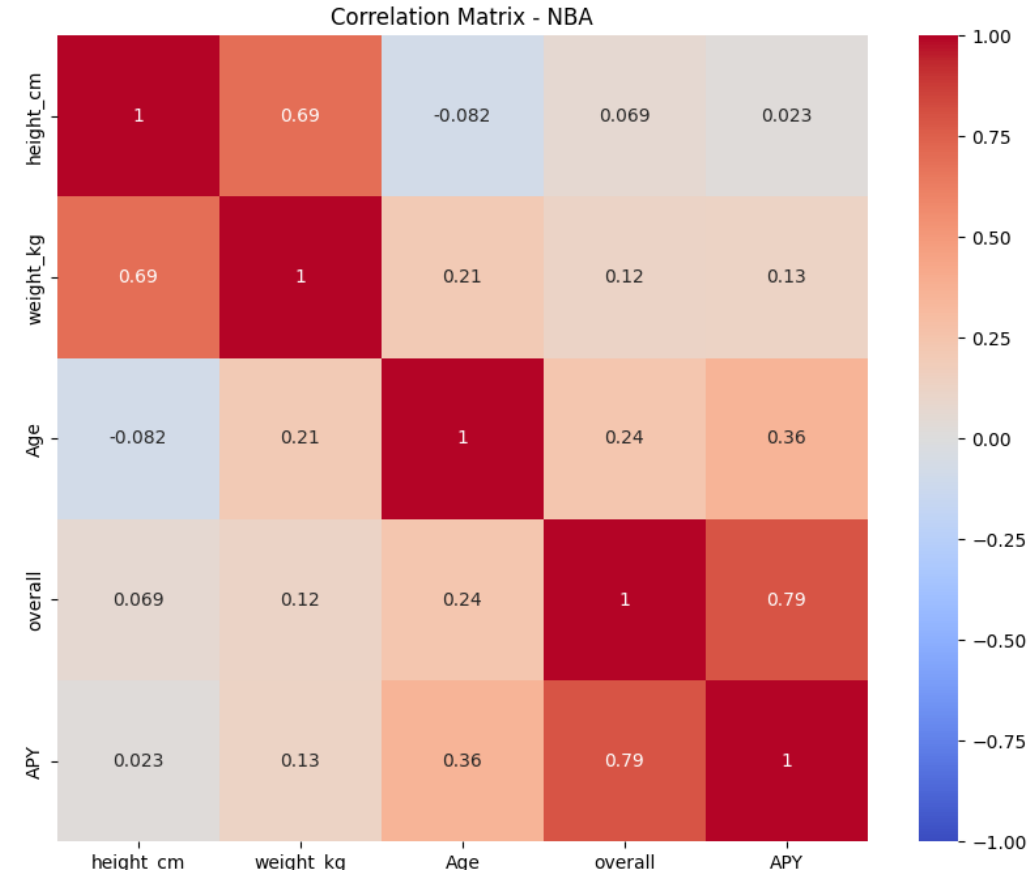


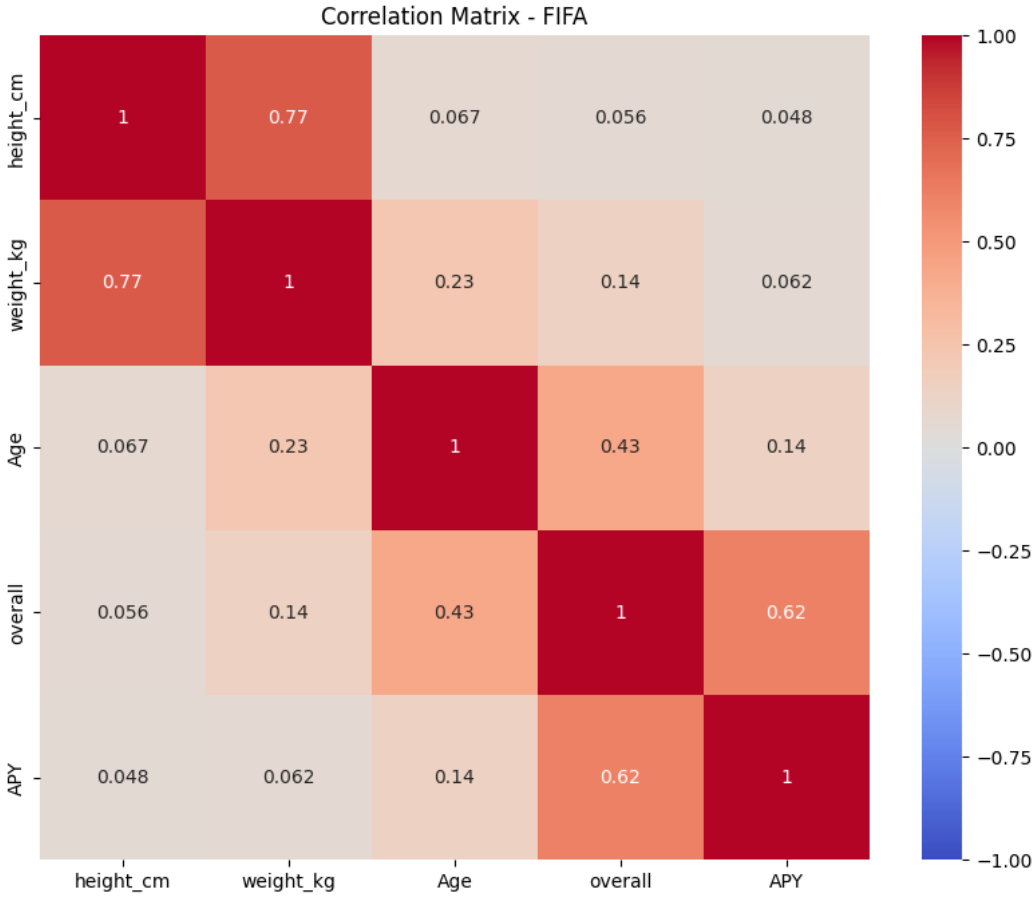
6. Correlation matrix

This visualization generates a heatmap for each sport showing how different attributes correlate with each other. The color intensity indicates the strength of the correlation (red for positive, blue for negative), while the numerical annotations show the exact correlation coefficients. This allows us to identify which factors have the strongest relationships with player performance and compensation in each sport, potentially revealing different value drivers across sports.

```
def plot_correlation_matrix(df):
    numeric_cols = ['height_cm', 'weight_kg', 'Age', 'overall', 'APY']
    for sport in df['sport'].unique():
        plt.figure(figsize=(10, 8))
        sport_data = df[df['sport'] == sport][numeric_cols]
        corr = sport_data.corr()
        sns.heatmap(corr, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
        plt.title(f'Correlation Matrix - {sport}')
        plt.show()
```




Correlation Matrix - NBA
height_cm
weight_kg
Age
overall
APY
1
0.69
-0.082
0.069
0.023
0.69
1
0.21
0.12
0.13
-0.082
0.21
1
0.24
0.36
0.069
0.12
0.24
1
0.79
0.023
0.13
0.36
0.79
1
height_cm
weight_kg
Age
overall
APY
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


Correlation Matrix - FIFA
height_cm
weight_kg
Age
overall
APY
1
0.77
0.067
0.056
0.048
0.77
1
0.23
0.14
0.062
0.067
0.23
1
0.43
0.14
0.056
0.14
0.43
1
0.62
0.048
0.062
0.14
0.62
1
height_cm
weight_kg
Age
overall
APY
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

7. Highest pay

The data helps us understand not only who the highest-paid athletes are but also what positions command top salaries in each sport. This information is particularly valuable for identifying patterns in how different sports value and compensate their top performers.

```
for sport in all_data['sport'].unique():
    print(f"\nTop 5 highest paid {sport} players:")
    top_players = all_data[all_data['sport'] == sport].nlargest(5, 'APY')
    print(top_players[['name', 'position', 'overall', 'APY']].to_string())
```

```
Top 5 highest paid NFL players:
              name position  overall         APY
0     Dak Prescott       QB       87  60000000.0
1       Joe Burrow       QB       95  55000000.0
2      Jordan Love       QB       70  55000000.0
3  Trevor Lawrence       QB       82  55000000.0
4   Tua Tagovailoa       QB       83  53100000.0

Top 5 highest paid NBA players:
               name position  overall         APY
1981  Stephen Curry       PG       95  55761216.0
1826    Joel Embiid        C       95  51415938.0
1928   Nikola Jokic        C       98  51415938.0
1863   Kevin Durant       PF       95  51179021.0
1664   Bradley Beal       SG       85  50203930.0

Top 5 highest paid FIFA players:
                                      name position  overall         APY
2642                       Kevin De Bruyne  CM, CAM       91  18200000.0
2423                     Robert Lewandowski       ST       90  17680000.0
7519                    Erling Braut Haaland       ST       91  17680000.0
7477  Vinícius José Paixão de Oliveira Júnior       LW       89  16120000.0
4576  Bernardo Mota Veiga de Carvalho e Silva  CM, RW       88  14040000.0
```

8. Average metrics by position for each sport

This analysis calculates the mean height, weight, overall rating, and annual pay for each position within each sport. The results reveal how different positions have evolved to optimize for specific physical attributes and how these specializations are reflected in player compensation. These positional averages help us understand the unique physical demands and value propositions of different roles within each sport.

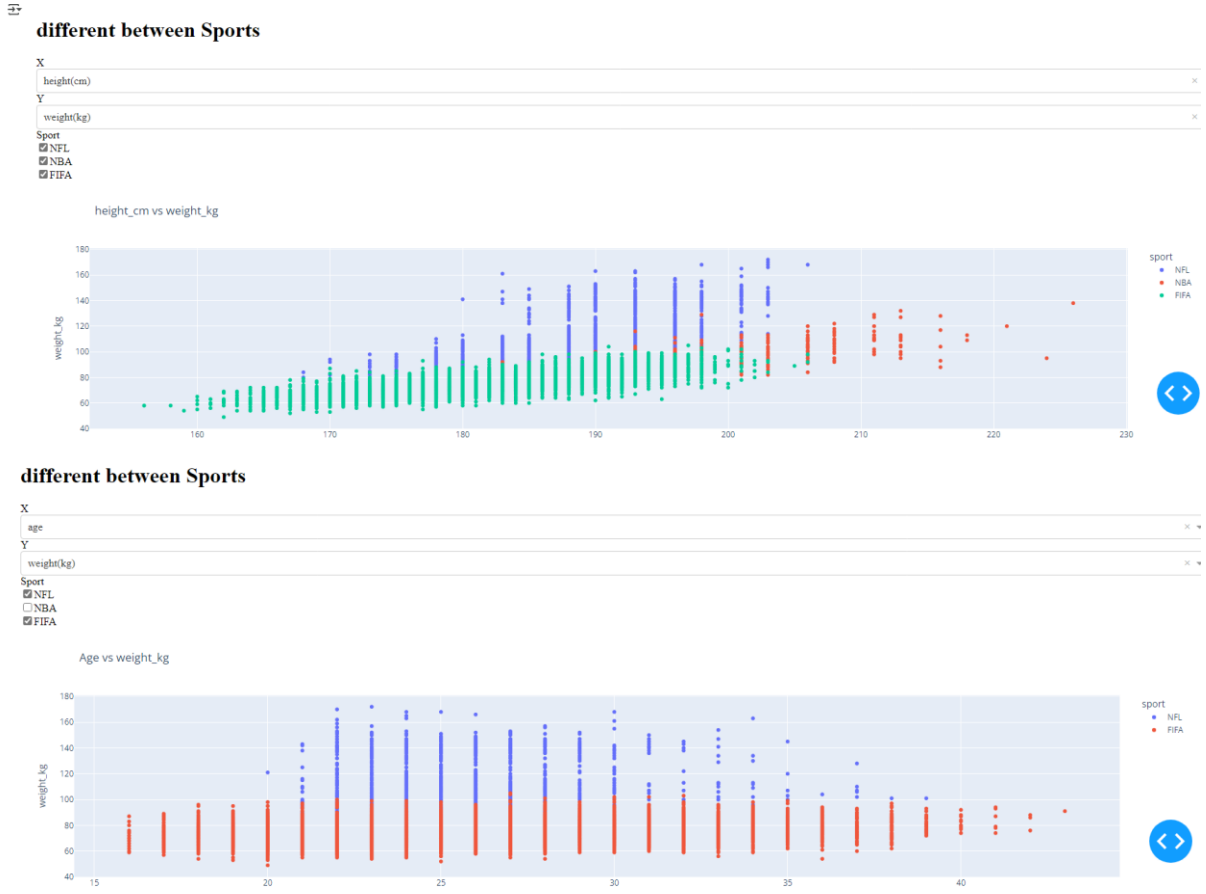
```
Average metrics by position for NFL:
          height_cm  weight_kg  overall          APY
position
C            192.70     139.30    69.74   3606781.98
CB           182.77      87.53    72.77   3520621.92
EDGE         192.80     118.24    72.16   5398534.03
FB           185.00     112.78    73.78   2359340.78
IDL          191.50     138.59    70.67   4817768.16
K            184.15      89.15    76.12   3531333.94
LB           186.73     106.02    70.45   3158581.08
LG           194.28     142.49    69.77   3931481.58
LS           188.07     108.97    33.59   1280510.52
LT           197.34     141.98    71.51   6266131.67
P            188.33      96.74    76.00   2142484.70
QB           189.00      99.04    68.07  12452613.23
RB           179.21      96.31    74.01   2825595.50
RG           194.48     144.28    70.22   4406197.96
RT           197.80     144.39    70.62   5963886.44
S            183.28      92.09    72.88   3372319.38
TE           194.57     112.42    70.31   3486221.17
WR           183.93      89.99    74.00   5333682.94

Average metrics by position for NBA:
          height_cm  weight_kg  overall          APY
position
C            209.98     110.20    78.85  12322152.34
PF           203.22     102.22    78.05  12429205.63
PG           190.81      88.05    79.04  15475753.64
SF           199.95      96.14    77.20  11594548.63
SG           195.30      91.16    77.89  11195754.46
```

9. Dashboard

Using Dash to explore relationships between different player attributes across sports.

The dashboard allows we to:

- Select any two metrics for comparison on the x and y axes
- Filter data by specific sports using checkboxes
- Automatically adjust to logarithmic scale when viewing salary data



5. Key insights

The correlation matrix analysis revealed an intriguing pattern: the correlation between age and annual pay (APY) is notably stronger than the correlations between physical attributes (height and weight) and compensation. This finding warrants further investigation in our research.

Additionally, we discovered a significant methodological consideration regarding salary comparisons across leagues. The different contract structures between FIFA (weekly wages) and NBA/NFL (annual contracts with guarantees) suggest that direct salary comparisons might not be the most effective approach. While incorporating guaranteed money into the APY calculations could provide more accurate comparisons, such data collection would require extensive database research.

Therefore, for future salary predictions, we propose focusing on relative compensation levels (e.g., top percentile earnings within each league) rather than

absolute monetary values. This approach would better account for the structural differences between leagues and provide more meaningful cross-sport comparisons.

6. Project timeline

FEB 21- MAR 2 Feature Engineering
MAR 2 -MAR 7 Feature Selection
MAR8 – MAR 14 Data Modeling
MAR14-20 REPORT