FIFA 2021
Database Analysis

Presented By:
Dan Boyer
Eric Ho
Paola Londono
Endalyn Oga

OBJECTIVE:

To analyze some of the different factors that could influence soccer players performance, such as wages amount, work load, contracts, and physical attributes.

SOURCE:

https://www.kaggle.com/

DISCUSSION TOPICS

- Wages vs. Performance
- Work Rate vs. Performance
- Contracts vs. Performance
- Physical Factors vs. Performance:
  - Weight
  - Age
  - Height

Data Cleaning

- Do the height and weight columns have the appropriate data types?

- Can you separate the joined column into three separate columns (year, month and day)?

- Can you clean and transform the value, wage and release clause columns into columns of integers?

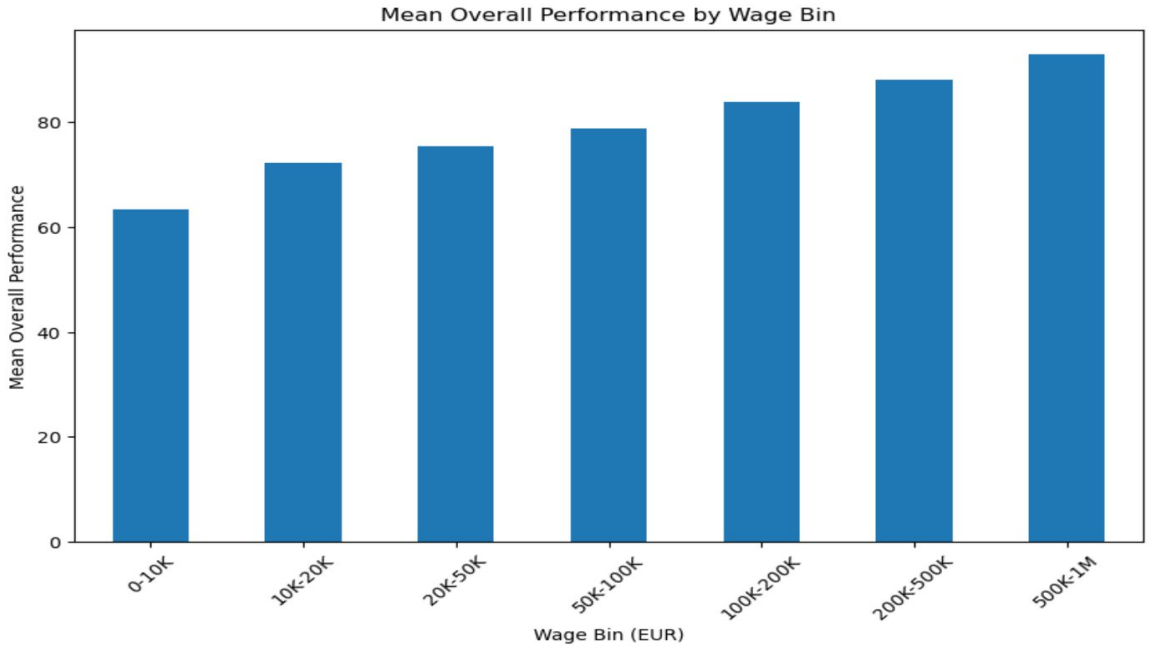
- How can you remove the newline characters from the Hits column?

- Should you separate the Team & Contract column into separate team and contract columns?

Analyze if higher wages correlate with better player performance ratings

Bar chart

It shows a clear upward trend, wages increase so does the predicted overall performance rating.

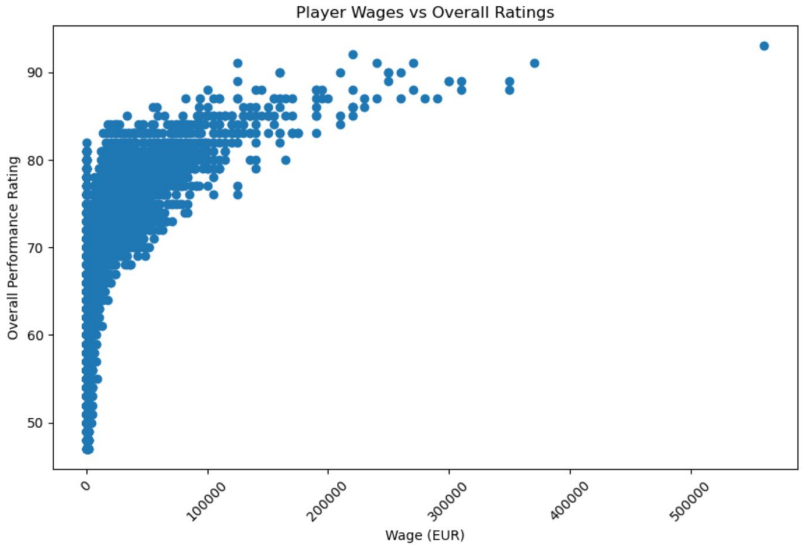


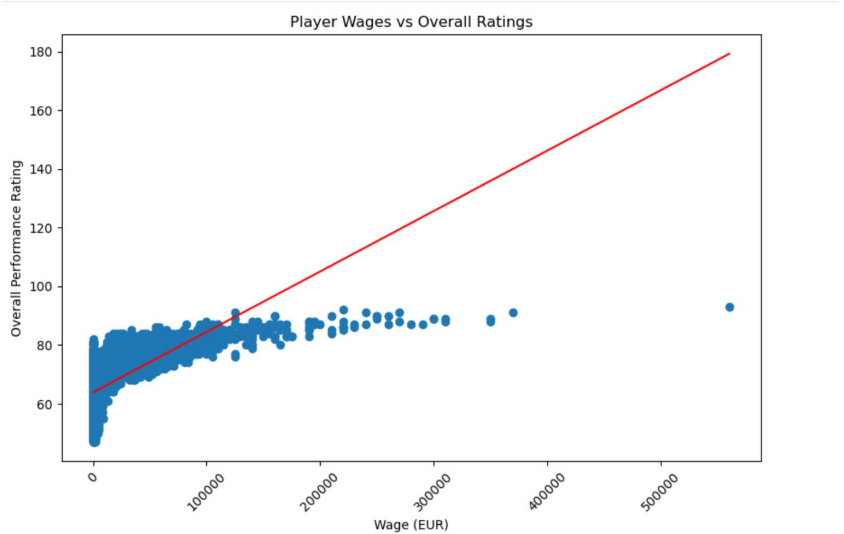
```python
# Define wage bins
wage_bins = [0, 10000, 20000, 50000, 100000, 200000, 500000, 1000000]  # Define wage ranges
wage_labels = ['0-10K', '10K-20K', '20K-50K', '50K-100K', '100K-200K', '200K-500K', '500K-1M']
```

```python
# Bin the wages
df_performance['wage_bin'] = pd.cut(df_performance['wage_eur'], bins=wage_bins, labels=wage_labels)
df_performance = df_performance.sort_values(by='wage_bin')
df_performance
```

Does higher wages correlate with better performance?

- **A correlation coefficient of 0.5809** indicates a moderate positive relationship between player wages and overall performance ratings.
- This suggests that, generally, **players with higher wages tend to have better performance ratings**. However, it's important to note that this correlation isn't perfect, meaning there are exceptions.
- Concentration of data
- Limited variability
- Potential outliers



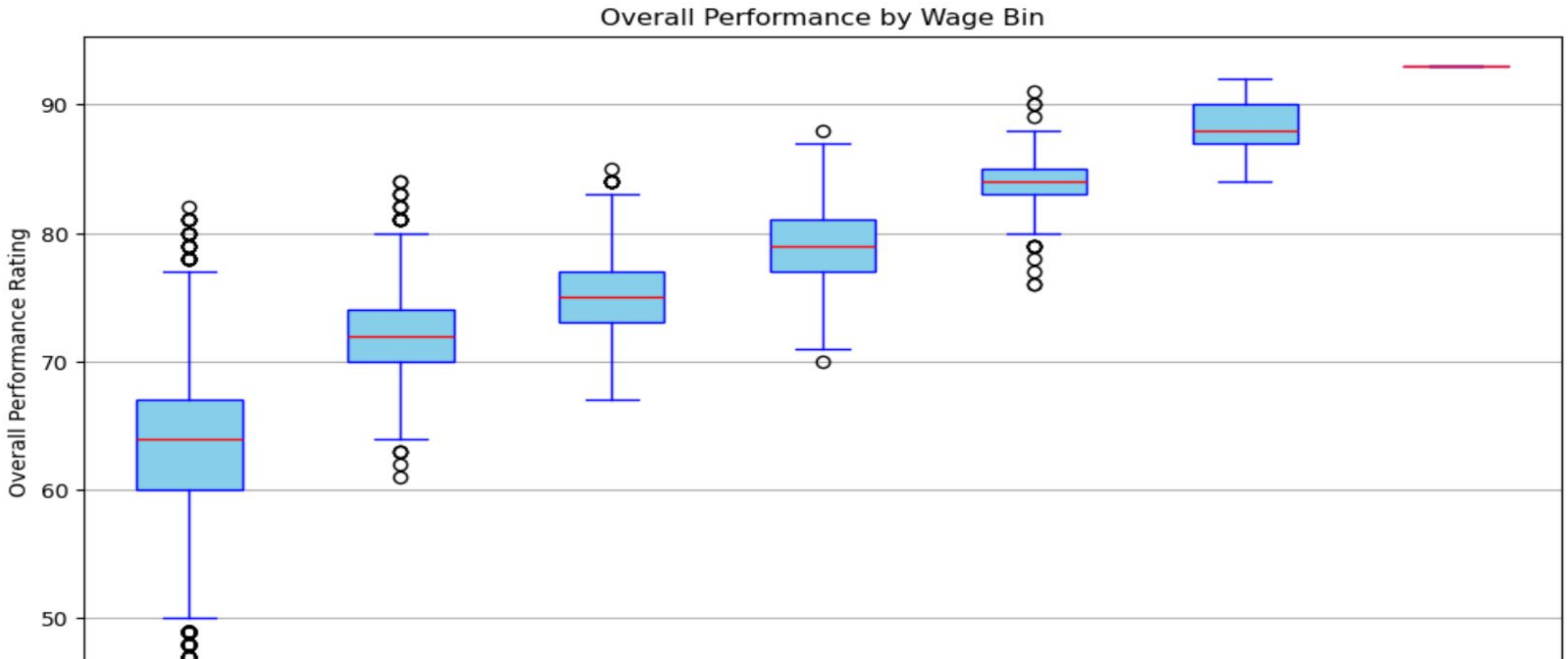


Statistic summary

- Comparing the mean overall ratings across the wage bins. The higher wage bins have significantly higher mean overall ratings, the players with higher wages generally perform better
- The median across wage bins follows the same pattern as the mean (increasing with wage), it supports the observation that higher wages are associated with better players.
- Variance for the lower wage bin 0-10k is high, this means there's a large spread in the overall ratings. It has mixed performance levels, with both high and low performers.

wage_bin	mean	median	var	std	sem
0-10K	63.381290	64.0	31.327610	5.597107	0.045754
10K-20K	72.164391	72.0	9.523899	3.086082	0.073603
20K-50K	75.369626	75.0	9.548917	3.090132	0.081376
50K-100K	78.896806	79.0	10.733167	3.276151	0.162393
100K-200K	83.880342	84.0	7.968317	2.822821	0.260970
200K-500K	88.103448	88.0	4.238916	2.058863	0.382321
500K-1M	93.000000	93.0	NaN	NaN	NaN

- We observed that higher wage bins tend to have higher medians and better overall ratings, but the spread (variance) of performance ratings also increases for higher wage bins.
- While higher wages are generally associated with better performance, some **lower-paid players still perform very well**, and also there are **higher-paid players with average ratings**.
- This variability suggests that while wage is a good indicator of performance, other factors also influence a player's rating.
- There are too many outliers in the box plot, and it seems the variance for the lower wage bin 0-10k is way too high. This means there's a large spread in the overall ratings. It has mixed performance levels, with both high and low performers.

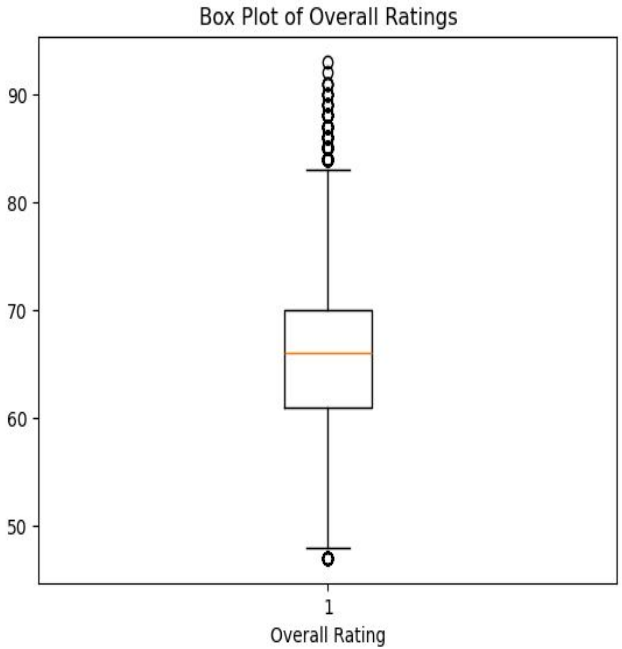


Summary:

Higher wages do correlate with better player performance ratings, but the relationship is not absolute. While most higher-wage players perform well, some exceptions exist, and performance variability is higher at the upper end of the wage spectrum. Wages are an important but not the sole factor in determining player performance. Teams and clubs may use this relationship as a guiding factor when deciding on player wages, but other qualitative factors should also be considered.

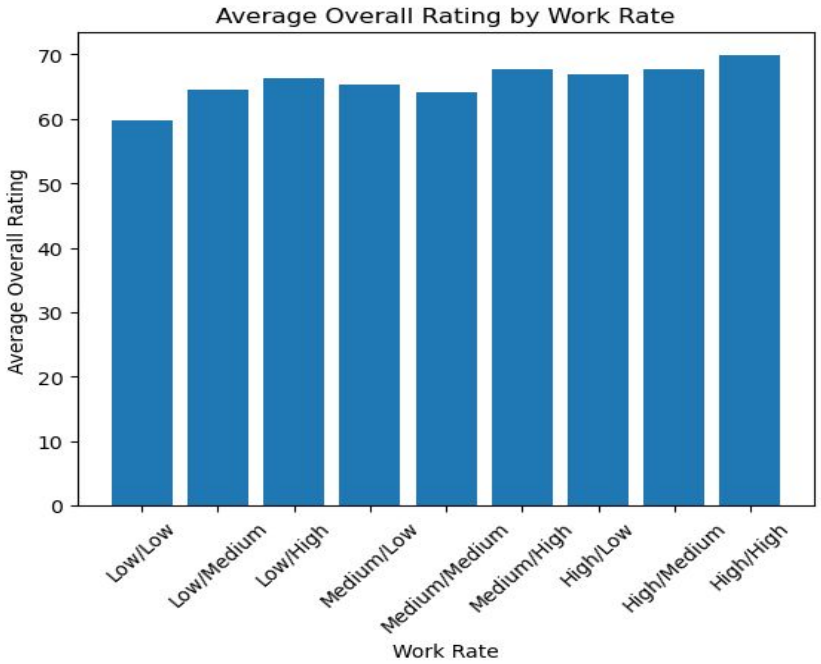
BOX PLOT OF OVERALL RATINGS

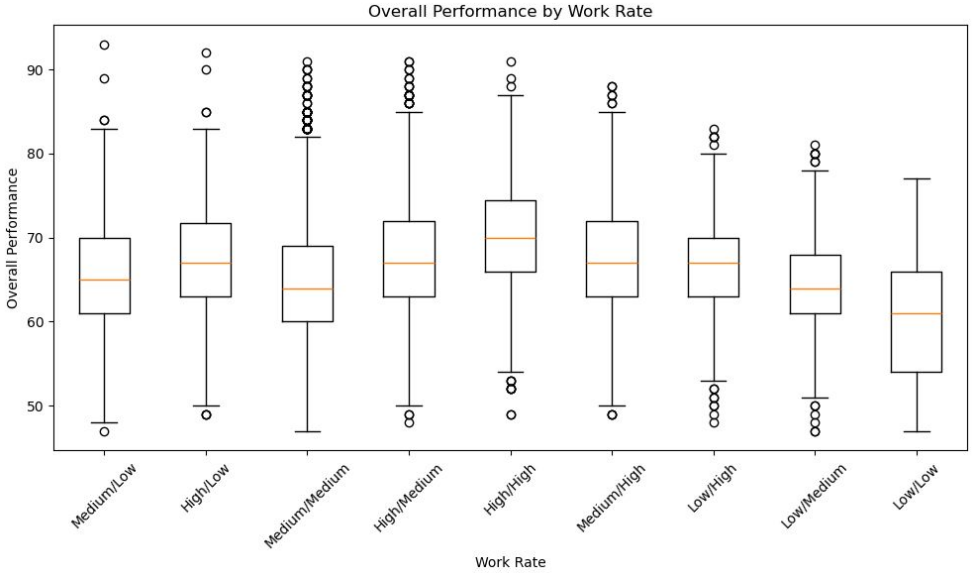
Box Pot of Overall Ratings



	Minimum	Q1 (25th Percentile)	Median (50th Percentile)	Q3 (75th Percentile)	Maximum
0	47	61.0	66.0	70.0	93

Overall Rating and Work Rate





ANOVA and Tukey Tests

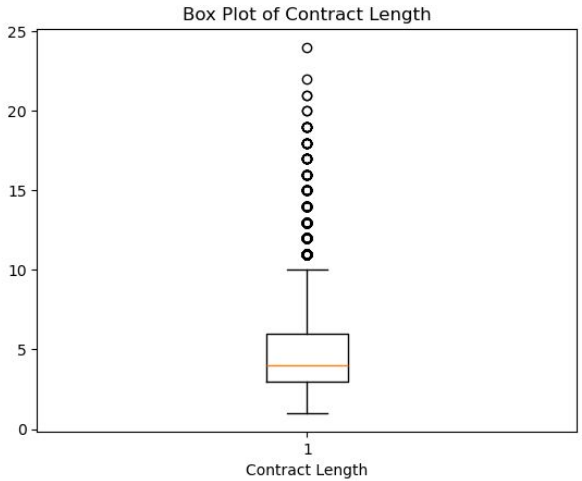
Anova test statistic = 180.893, P- value = pvalue=7.803327969335649e-296

Multiple Comparison of Means - Tukey HSD, FWER=0.05

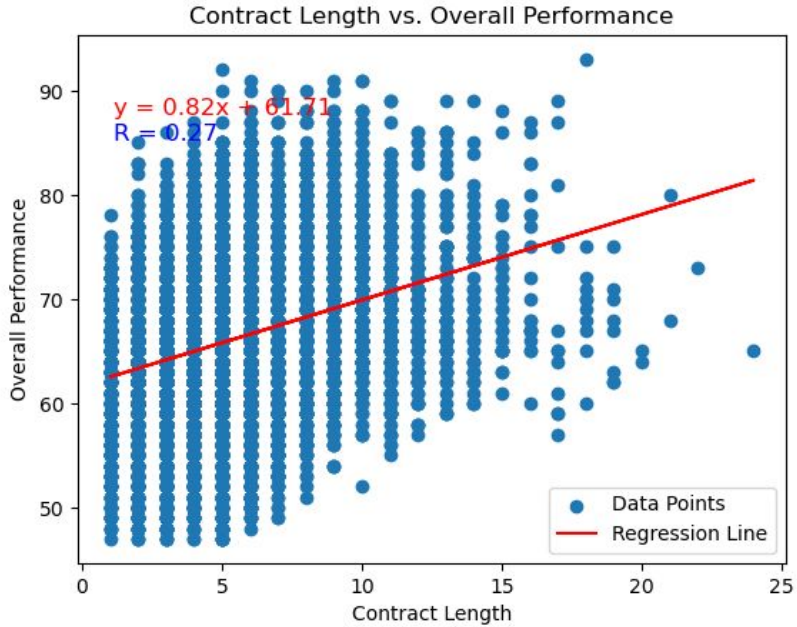
group1	group2	meandiff	p-adj	lower	upper	reject
High/High	High/Low	-2.9987	0.0	-3.9903	-2.0071	True
High/High	High/Medium	-2.1195	0.0	-2.8626	-1.3763	True
High/High	Low/High	-3.5564	0.0	-4.7432	-2.3695	True
High/High	Low/Low	-10.0688	0.0	-13.0183	-7.1194	True
High/High	Low/Medium	-5.4298	0.0	-6.5852	-4.2743	True
High/High	Medium/High	-2.2234	0.0	-3.0414	-1.4053	True
High/High	Medium/Low	-4.5765	0.0	-5.5282	-3.6249	True
High/High	Medium/Medium	-5.6773	0.0	-6.3624	-4.9923	True
High/Low	High/Medium	0.8792	0.0274	0.0517	1.7068	True
High/Low	Low/High	-0.5577	0.9007	-1.7991	0.6838	False
High/Low	Low/Low	-7.0701	0.0	-10.042	-4.0983	True
High/Low	Low/Medium	-2.4311	0.0	-3.6425	-1.2196	True
High/Low	Medium/High	0.7753	0.1527	-0.1201	1.6708	False
High/Low	Medium/Low	-1.5778	0.0001	-2.5967	-0.5589	True
High/Low	Medium/Medium	-2.6786	0.0	-3.4544	-1.9028	True
High/Medium	Low/High	-1.4369	0.0008	-2.4905	-0.3833	True
High/Medium	Low/Low	-7.9494	0.0	-10.8478	-5.051	True
High/Medium	Low/Medium	-3.3103	0.0	-4.3285	-2.2921	True
High/Medium	Medium/High	-0.1039	0.9998	-0.7128	0.5051	False
High/Medium	Medium/Low	-2.457	0.0	-3.2363	-1.6778	True
High/Medium	Medium/Medium	-3.5579	0.0	-3.9712	-3.1445	True
Low/High	Low/Low	-6.5125	0.0	-9.555	-3.4699	True
Low/High	Low/Medium	-1.8734	0.0008	-3.2493	-0.4975	True
Low/High	Medium/High	1.333	0.0059	0.2252	2.4408	True
Low/High	Medium/Low	-1.0202	0.1798	-2.2299	0.1896	False
Low/High	Medium/Medium	-2.121	0.0	-3.1345	-1.1075	True
Low/Low	Low/Medium	4.6391	0.0001	1.6086	7.6695	True
Low/Low	Medium/High	7.8455	0.0	4.927	10.764	True
Low/Low	Medium/Low	5.4923	0.0	2.5336	8.4511	True
Low/Low	Medium/Medium	4.3915	0.0001	1.5074	7.2756	True
Low/Medium	Medium/High	3.2064	0.0	2.1323	4.2805	True
Low/Medium	Medium/Low	0.8532	0.3764	-0.3258	2.0323	False
Low/Medium	Medium/Medium	-0.2476	0.9972	-1.2242	0.729	False
Medium/High	Medium/Low	-2.3532	0.0	-3.2042	-1.5022	True
Medium/High	Medium/Medium	-3.454	0.0	-3.9905	-2.9175	True
Medium/Low	Medium/Medium	-1.1008	0.0001	-1.8249	-0.3768	True

Contract Length Boxplot

	Minimum	Q1 (25th Percentile)	Median (50th Percentile)	Q3 (75th Percentile)	Maximum
0	1.0	3.0	4.0	6.0	24.0



Contract vs Performance



R^2: 0.0708413166222214

PHYSICAL FACTORS VS. PERFORMANCE

How do Weight and Age affect soccer players performance?

By Paola Londono

Physical Data vs Performance Data

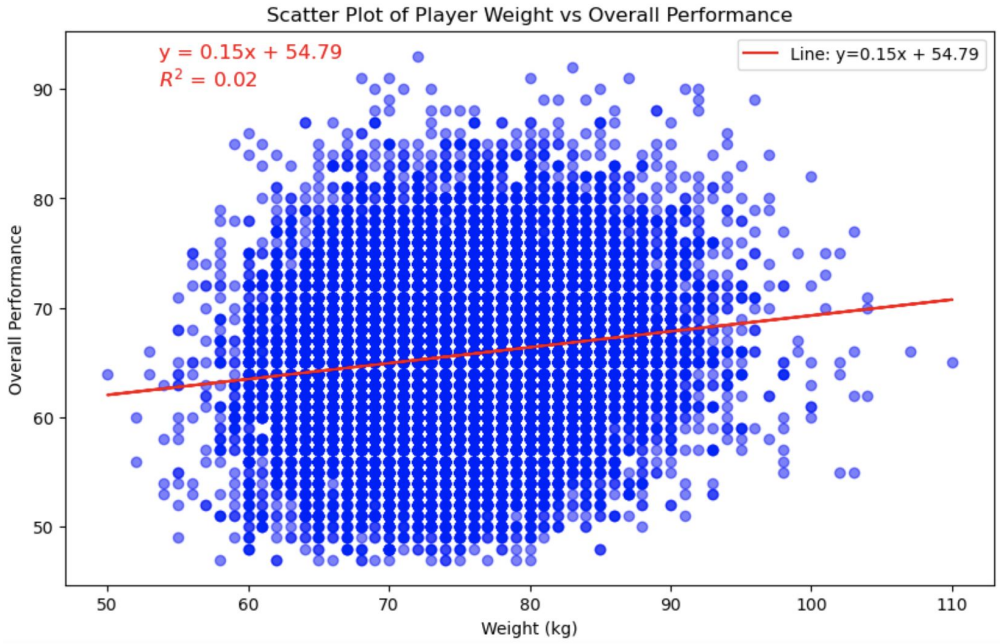
Height **Weight** **Age**	Pace Shooting Passing Dribbling Defending **Overall**

PERFORMANCE IN RELATION TO WEIGHT

```
# Create a data frame for weight and overall performance for each player
df_weight = df.set_index('short_name')[['weight_kg', 'overall']]
df_weight
```

short_name	weight_kg	overall
L. Messi	72	93
Cristiano Ronaldo	83	92
J. Oblak	87	91
R. Lewandowski	80	91
Neymar Jr	68	91
...	...	...
K. Angulo	73	47
Zhang Mengxuan	70	47
Wang Zhenghao	74	47
Chen Zitong	80	47
Song Yue	79	47

18944 rows × 2 columns

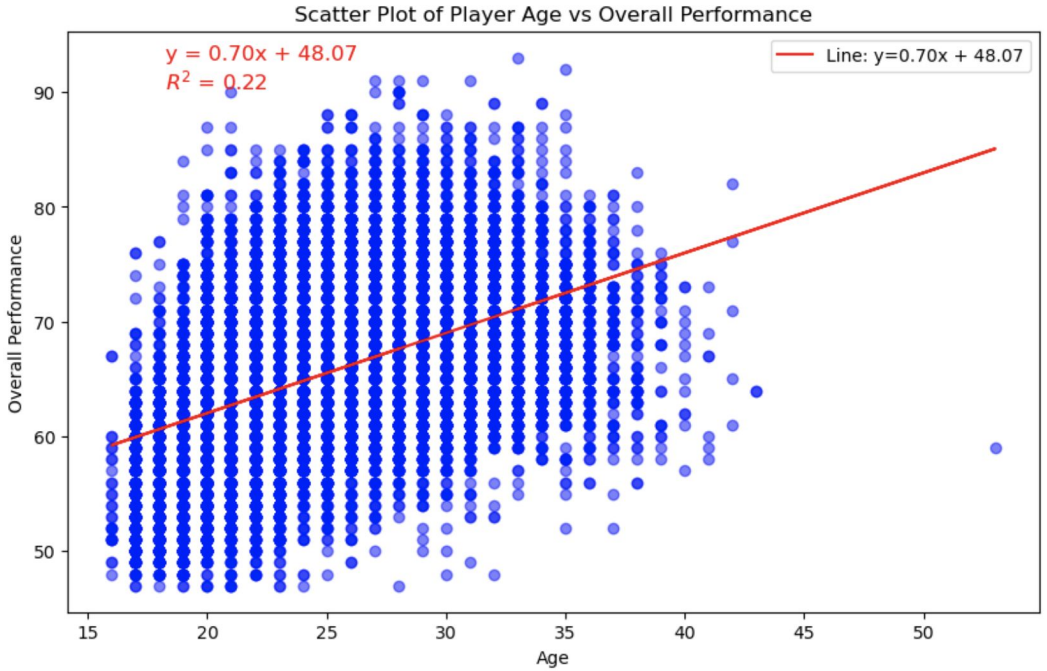


PERFORMANCE IN RELATION TO AGE

```
# Create a data frame for age and overall performance for each player
df_age = df.set_index('short_name')[['age', 'overall']]
df_age
```

short_name	age	overall
L. Messi	33	93
Cristiano Ronaldo	35	92
J. Oblak	27	91
R. Lewandowski	31	91
Neymar Jr	28	91
...	...	...
K. Angulo	24	47
Zhang Mengxuan	21	47
Wang Zhenghao	20	47
Chen Zitong	23	47
Song Yue	28	47

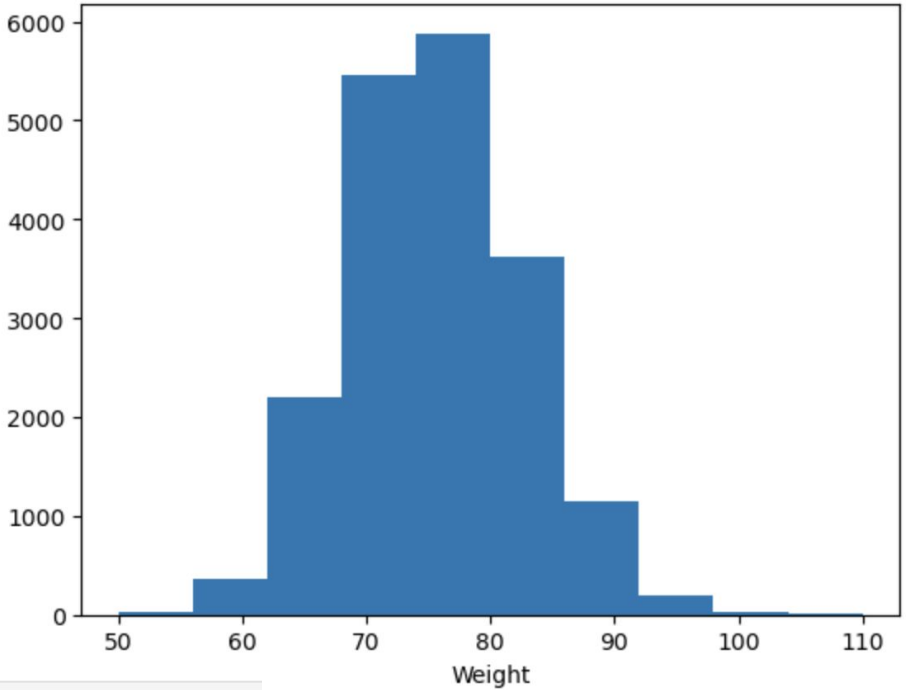
18944 rows × 2 columns



WEIGHT ANALYSIS

```
#Find mean, median and mode for Weight:
Wmean = np.mean(df_weight['weight_kg'])
Wmedian = np.median(df_weight['weight_kg'])
Wmode = st.mode(df_weight['weight_kg'])
print(f"mean = {Wmean}")
print(f"median = {Wmedian}")
print(f"mode = {Wmode}")
```

```
mean = 75.01689189189189
median = 75.0
mode = ModeResult(mode=70, count=1510)
```




```
#Test for normality
print(st.normaltest(df_weight['weight_kg'].sample(500)))
```

```
NormaltestResult(statistic=0.24353699537795603, pvalue=0.8853533061163831)
```

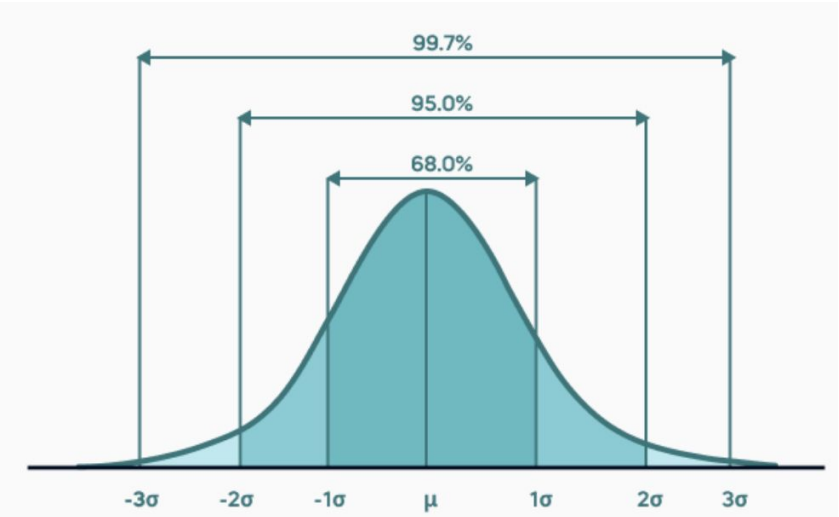
WEIGHT ANALYSIS

```
#Calculate Variance and Stantard Deviation for Weight
WVariance = np.var(df_weight['weight_kg'])
Wstd = np.std(df_weight['weight_kg'])
print(f"Variance: {WVariance}, Standard Deviation: {Wstd}")
```

```
Variance: 49.800601488312644, Standard Deviation: 7.056954122588062
```

```
#Find the Standard deviations distribution along the weight curve
wstd_minus_1 = round(Wmean - Wstd,3)
wstd_minus_2 = round(Wmean - 2 * Wstd,3)
wstd_minus_3 = round(Wmean - 3 * Wstd,3)
wstd_plus_1 = round(Wmean + Wstd,3)
wstd_plus_2 = round(Wmean + 2 * Wstd,3)
wstd_plus_3 = round(Wmean + 3 * Wstd,3)
print("std-1:", wstd_minus_1)
print("std-2:", wstd_minus_2)
print("std-3:", wstd_minus_3)
print("std+1:", wstd_plus_1)
print("std+2:", wstd_plus_2)
print("std+3:", wstd_plus_3)
```

```
std-1: 67.96
std-2: 60.903
std-3: 53.846
std+1: 82.074
std+2: 89.131
std+3: 96.188
```



https://365datascience.com/calculators/standard-deviation-calculator/

WEIGHT ANALYSIS

```
#Identify Outliers:

Wquartiles = df_weight['weight_kg'].quantile([.25,.5,.75])
Wlowerq = Wquartiles[.25]
Wmedian = Wquartiles[.5]
Wupperq = Wquartiles[.75]
print(f"Lower Quartile: {Wlowerq}, Median: {Wmedian}, Upper Quartile: {Wupperq}")
weight_IQR = Wupperq - Wlowerq
print(f"Weight IQR: {weight_IQR}")
w_lower_bound = Wlowerq - 1.5 * weight_IQR
w_upper_bound = Wupperq + 1.5 * weight_IQR
print(f"Weight Lower Bound: {w_lower_bound}, Weight Upper bound: {w_upper_bound}")
```

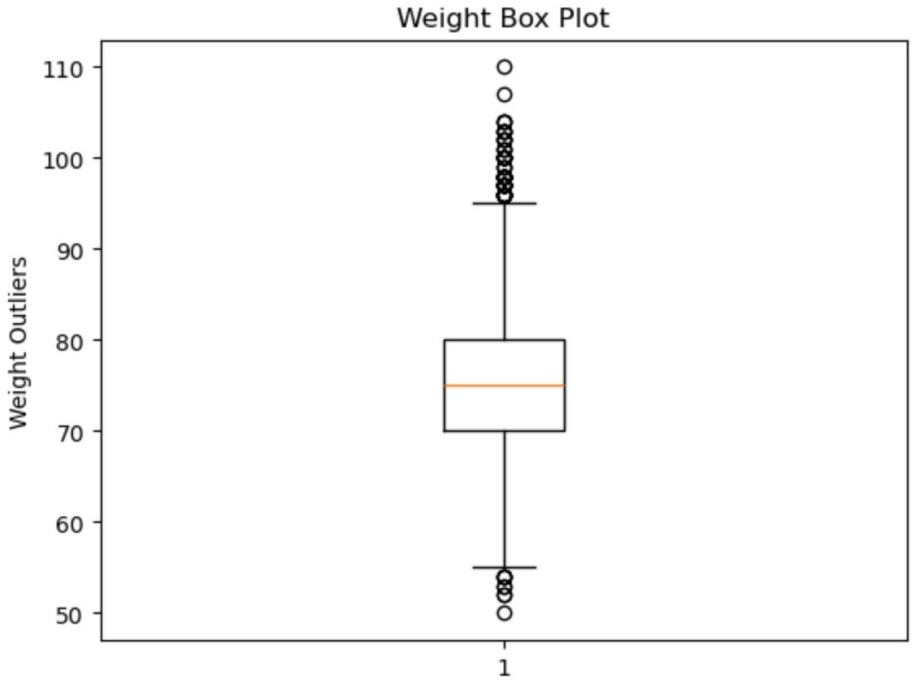
```
Lower Quartile: 70.0, Median: 75.0, Upper Quartile: 80.0
Weight IQR: 10.0
Weight Lower Bound: 55.0, Weight Upper bound: 95.0
```

```
#Maximum weight
df_weight['weight_kg'].max()
```

```
110
```

```
#Minimum weight
df_weight['weight_kg'].min()
```

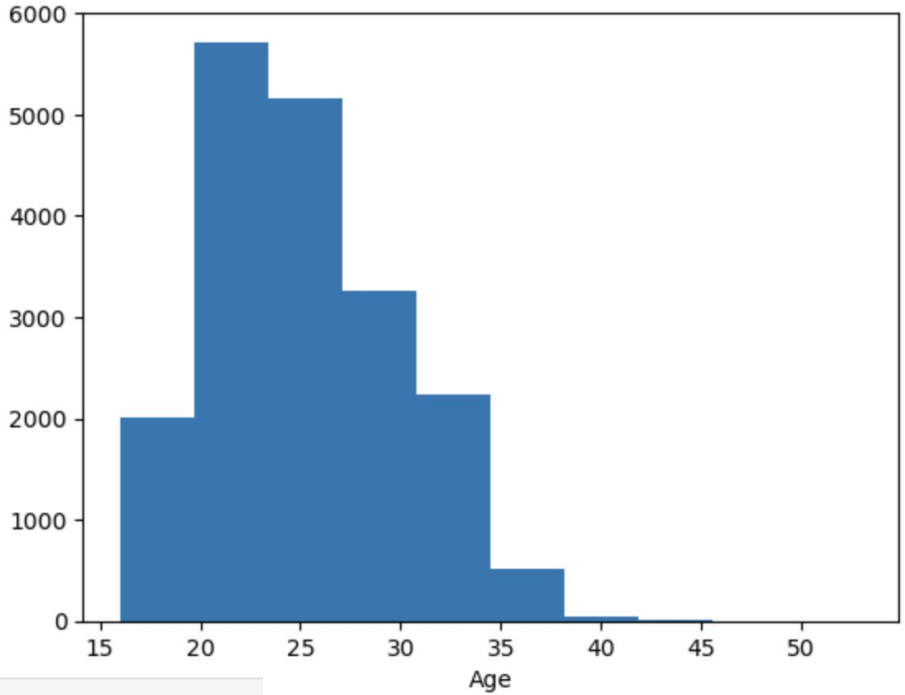
```
50
```



AGE ANALYSIS

```
#Find mean, median and mode for Age:
Amean = np.mean(df_age['age'])
Amedian = np.median(df_age['age'])
Amode = st.mode(df_age['age'])
print(f"mean = {Amean}")
print(f"median = {Amedian}")
print(f"mode = {Amode}")
```

```
mean = 25.22582347972973
median = 25.0
mode = ModeResult(mode=23, count=1473)
```




```
#Test for normality
print(st.normaltest(df_age['age'].sample(500)))
```

```
NormaltestResult(statistic=18.37670073213996, pvalue=0.00010222335681212253)
```

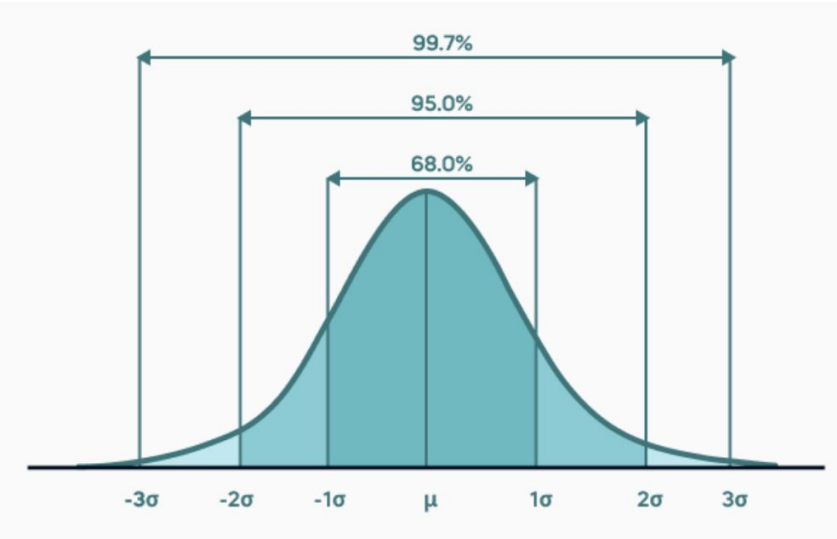
AGE ANALYSIS

```
#Calculate Variance and Stantard Deviation for Weight
AVariance = np.var(df_age['age'])
Astd = np.std(df_age['age'])
print(f"Variance: {AVariance}, Standard Deviation: {Astd}")
```

```
Variance: 22.063974195191946, Standard Deviation: 4.697230481378569
```

```
#Find the Standard deviations distribution along the  age curve
astd_minus_1 = round(Amean - Astd,3)
astd_minus_2 = round(Amean - 2 * Astd,3)
astd_minus_3 = round(Amean - 3 * Astd,3)
astd_plus_1 = round(Amean + Astd,3)
astd_plus_2 = round(Amean + 2 * Astd,3)
astd_plus_3 = round(Amean + 3 * Astd,3)
print("std-1:", astd_minus_1)
print("std-2:", astd_minus_2)
print("std-3:", astd_minus_3)
print("std+1:", astd_plus_1)
print("std+2:", astd_plus_2)
print("std+3:", astd_plus_3)
```

```
std-1: 20.529
std-2: 15.831
std-3: 11.134
std+1: 29.923
std+2: 34.62
std+3: 39.318
```



https://365datascience.com/calculators/standard-deviation-calculator/

AGE ANALYSIS

```
#Identify Outliers:

Aquartiles = df_age['age'].quantile([.25,.5,.75])
Alowerq = Aquartiles[.25]
Amedian = Aquartiles[.5]
Aupperq = Aquartiles[.75]
print(f"Lower Quartile: {Alowerq}, Median: {Amedian}, Upper Quartile: {Aupperq}")
age_IQR = Aupperq - Alowerq
print(f"Age IQR: {age_IQR}")
a_lower_bound = Alowerq - 1.5 * age_IQR
a_upper_bound = Aupperq + 1.5 * age_IQR
print(f"Age Lower Bound: {a_lower_bound}, Age Upper bound: {a_upper_bound}")
```

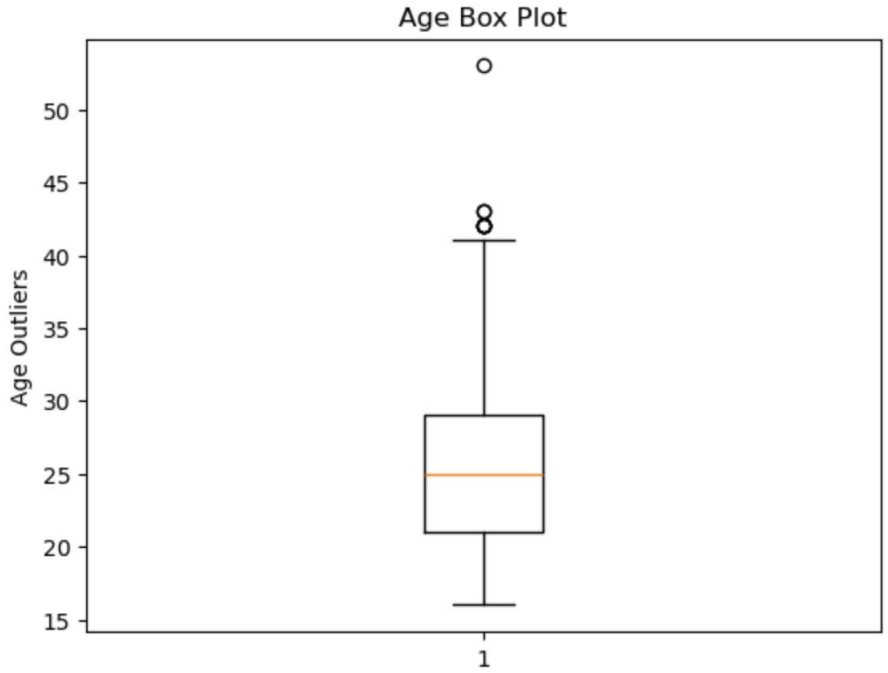
```
Lower Quartile: 21.0, Median: 25.0, Upper Quartile: 29.0
Age IQR: 8.0
Age Lower Bound: 9.0, Age Upper bound: 41.0
```

```
#Maximum age
df_age['age'].max()
```

```
53
```

```
#Minimum age
df_age['age'].min()
```

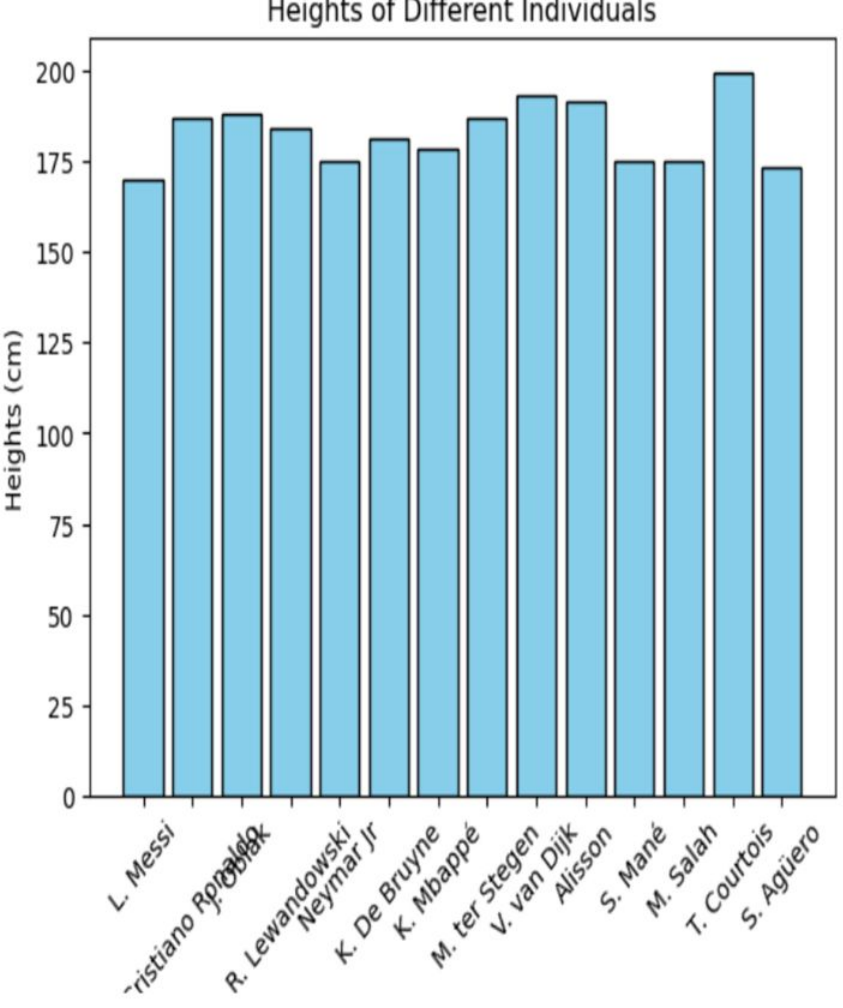
```
16
```



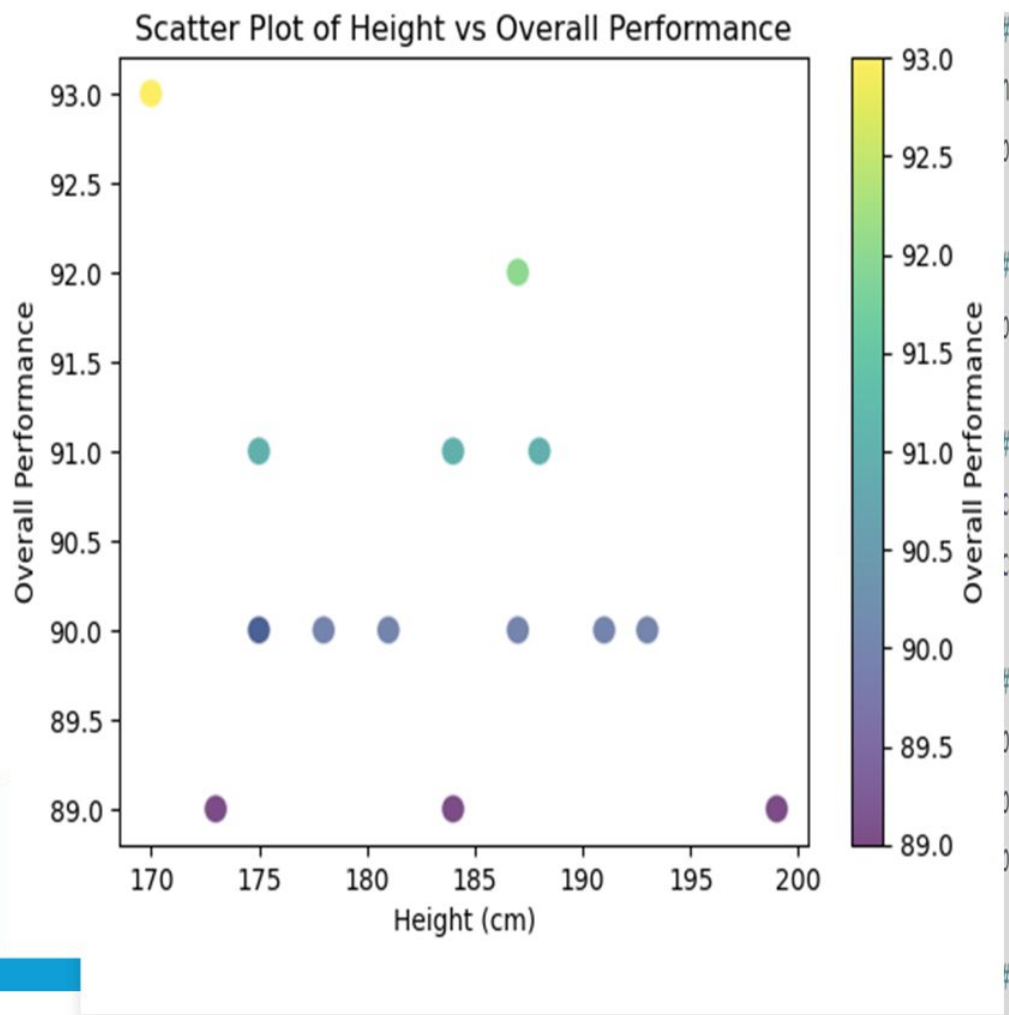
PERFORMANCE IN RELATION TO HEIGHT

 # data =

- { 'Height': [170, 187, 188, 184, 175, 181, 178, 187, 193, 191, 175, 175, 199, 173, 184],
- 'Overall_performance': [93, 92, 91, 91, 91, 90, 90, 90, 90, 90, 90, 90, 89, 89, 89]




```python
import matplotlib.pyplot as plt

# Data
short_name = ['L. Messi', 'Cristiano Ronaldo', 'J. Oblak', 'R. Lewandowski', 'Neymar Jr', 'K. De Bruyne', 'K. Mbappé', 'M. ter Stegen', 'V. van Dijk',
'Alisson', 'S. Mané', 'M. Salah', 'T. Courtois', 'S. Agüero']
height_cm = [170, 187, 188, 184, 175, 181, 178, 187, 193, 191, 175, 175, 199, 173]

# Create bar plot
plt.bar(short_name, height_cm, color='skyblue', edgecolor='black')

# Set labels and title
plt.xlabel('Names')
plt.ylabel('Heights (cm)')
plt.title('Heights of Different Individuals')

# Rotate x-axis labels
plt.xticks(rotation=45)

# Show plot
plt.show()
```




```python
# Convert data to numpy arrays for easier handling
heights = np.array(data['Height'])
performance = np.array(data['Overall_performance'])

# Create scatter plot
plt.scatter(heights, performance, c=performance, cmap='viridis', s=100, alpha=0.7, edgecolors='w')

# Add color bar
cbar = plt.colorbar()
cbar.set_label('Overall Performance')

# Add labels and title
plt.xlabel('Height (cm)')
plt.ylabel('Overall Performance')
plt.title('Scatter Plot of Height vs Overall Performance')

# Show plot
plt.show()
```

CONCLUSION

- While various factors influence player performance, no single factor dominates.
- Two of the stronger correlations found through regression analysis are work rate and wage.
- A higher work rate is generally associated with better performance, although there are exceptions.
- Regarding wage, it is suggested that players with better performance are likely to earn more, rather than higher pay directly improving performance.
- Proposal to extending the study by tracking player performance over time to observe changes and eliminate comparison errors between players with inherently different skill levels.