SCOUTIFY

Sooraj Sathish (IMT2020004)
Kritin Potluru (IMT2020027)
Chinmay Parekh (IMT2020069)
Keshav Goyal (IMT2020101)

Note: According to presentation feedback, a richer analysis has been done in the report.

Motivation:A Story of Two Players

André Schürrle

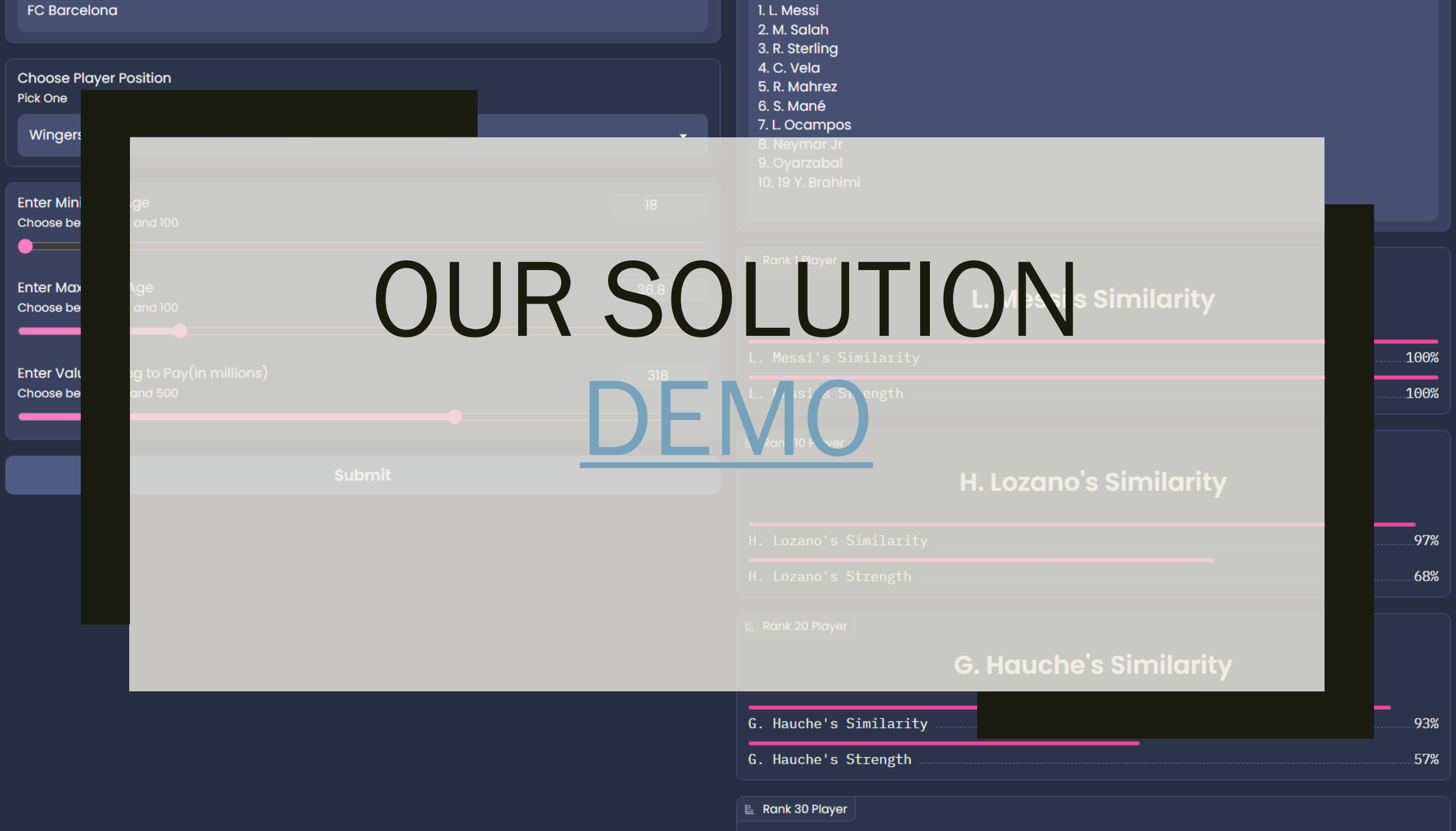
- Valued at £20-30 million
- Overall rating of 70-80
- Scored once in 2-3 matches
- Sold to Borussia Dortmund and had a lackluster career

Mo Salah

- Valued at £20-30 million
- Overall rating of 70-80
- Scored once in 2-3 matches
- Sold to Liverpool, 1 UEFA Champions League title, 2 Premier League titles, Premier League Golden Boot and PFA Player of the Year

The Requirement

- One thing we can do is to base our assessments on **more objective data.** If we can design and validate systems to learn patterns that are reliably associated with subsequent high performance, and prove that they work in the real world, we can start to make better decisions and take human bias out of the equation.
- To create a recommender system that connects clubs and players
- System must match clubs and players and such matches must be mutually beneficial
- System should keep in mind:
  - Club Preferences
  - Player Preferences
  - Budget of Clubs


FC Barcelona
1. L. Messi
2. M. Salah
3. R. Sterling
4. C. Vela
5. R. Mahrez
6. S. Mané
7. L. Ocampos
8. Neymar Jr
9. Oyarzabal
10. 19 Y. Brahimi
Choose Player Position
Pick One
Wingers
Enter Mini
Choose be
Enter Max
Choose be
Enter Valu
Choose be
ge
and 100
18
Age
and 100
36.8
g to Pay(in millions)
and 500
318
Submit
OUR SOLUTION
DEMO
Rank 1 Player
L. Messi's Similarity
L. Messi's Similarity
100%
L. Messi's Strength
100%
Rank 10 Player
H. Lozano's Similarity
H. Lozano's Similarity
97%
H. Lozano's Strength
68%
Rank 20 Player
G. Hauche's Similarity
G. Hauche's Similarity
93%
G. Hauche's Strength
57%
Rank 30 Player

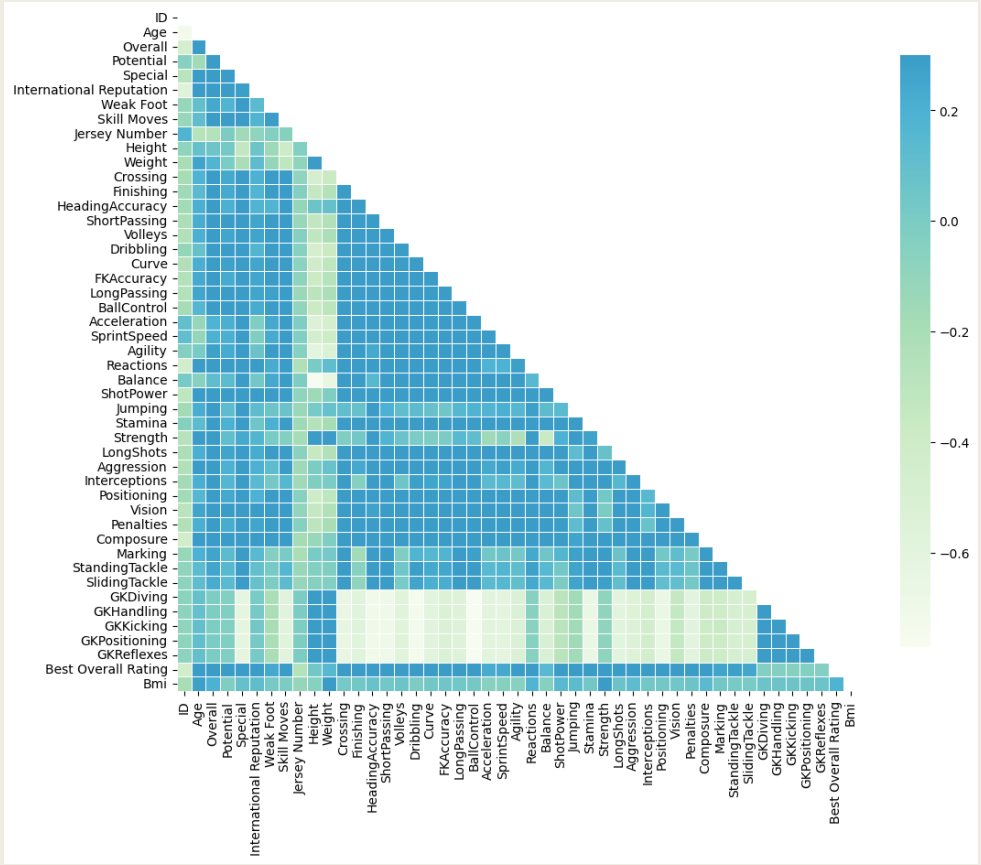
Dataset

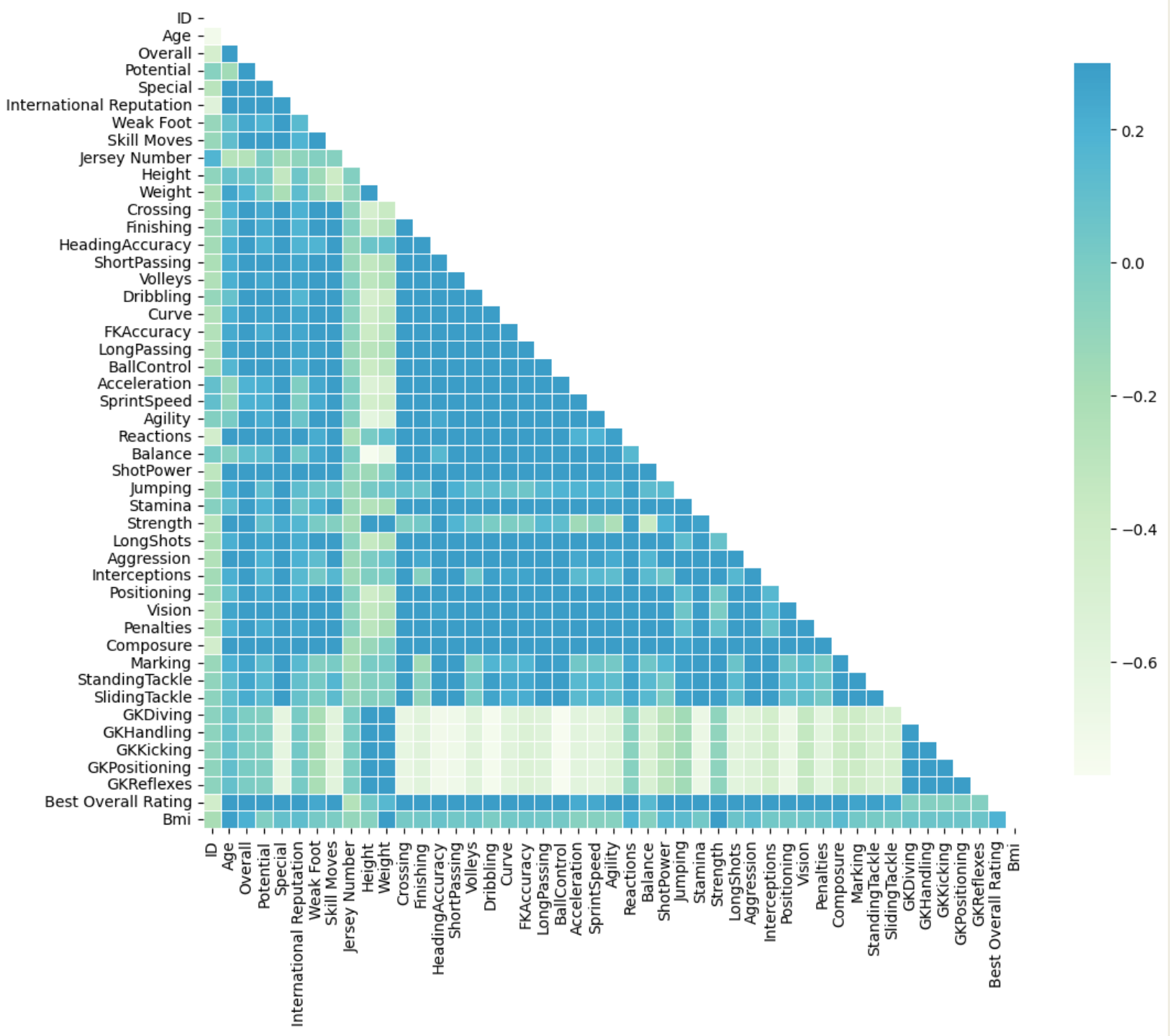
	ID	Name	Age	Nationality	Overall	Potential	Club	Value	Weak Foot	Skill Moves	...	SlidingTackle	GKDiving
0	176580	L. Suárez	29	Uruguay	92	92	FC Barcelona	€83M	4.0	4.0	...	38.0	27.0
1	178518	R. Nainggolan	28	Belgium	86	86	Roma	€37.5M	3.0	3.0	...	88.0	11.0
2	181872	A. Vidal	29	Chile	87	87	FC Bayern München	€41.5M	4.0	3.0	...	84.0	4.0
3	197445	D. Alaba	24	Austria	86	89	FC Bayern München	€41.5M	4.0	3.0	...	83.0	5.0
4	195864	P. Pogba	23	France	88	94	Manchester United	€71.5M	4.0	5.0	...	73.0	5.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
104347	240558	18 L. Clayton	17	England	53	70	Cheltenham Town	€100K	2.0	1.0	...	12.0	55.0
104348	262846	. Dobre	20	Romania	53	63	FC Academica Clinceni	€180K	2.0	1.0	...	12.0	57.0
104349	241317	21 Xue Qinghao	19	China PR	47	60	Shanghai Shenhua FC	€100K	2.0	1.0	...	9.0	49.0
104350	259646	A. Shaikh	18	India	47	67	ATK Mohun Bagan FC	€110K	3.0	1.0	...	13.0	49.0

- Every player available in FIFA 17,18, 19, 20, 21 and 22
- 104 Features
- Player positions, with the role in the club and in the national team
- Player attributes with statistics as Attacking, Skills, Defense, Mentality, GK Skills, etc.
- Player personal data like Nationality, Club, DateOfBirth, Wage, Salary, etc.

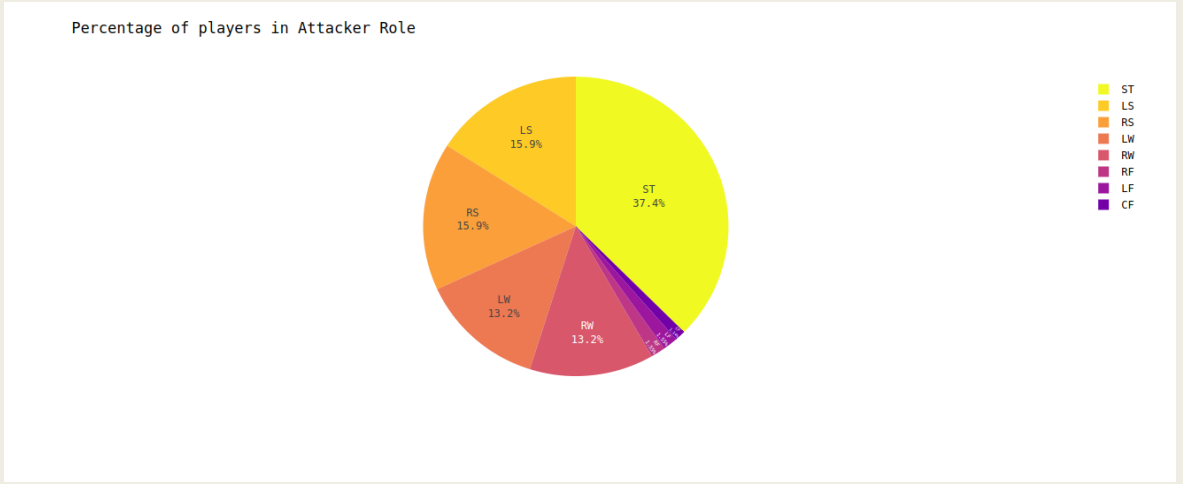
Dataset – EDA : Validity of Data

- From this correlation chart, we can see that Goalkeeper's attributes have a strong negative correlation with the attributes possessed by an outfield player.
- Players with more attacking role have higher dribbling, passing and shooting skills.
- There are numerous such real-world patterns seen in the chart.

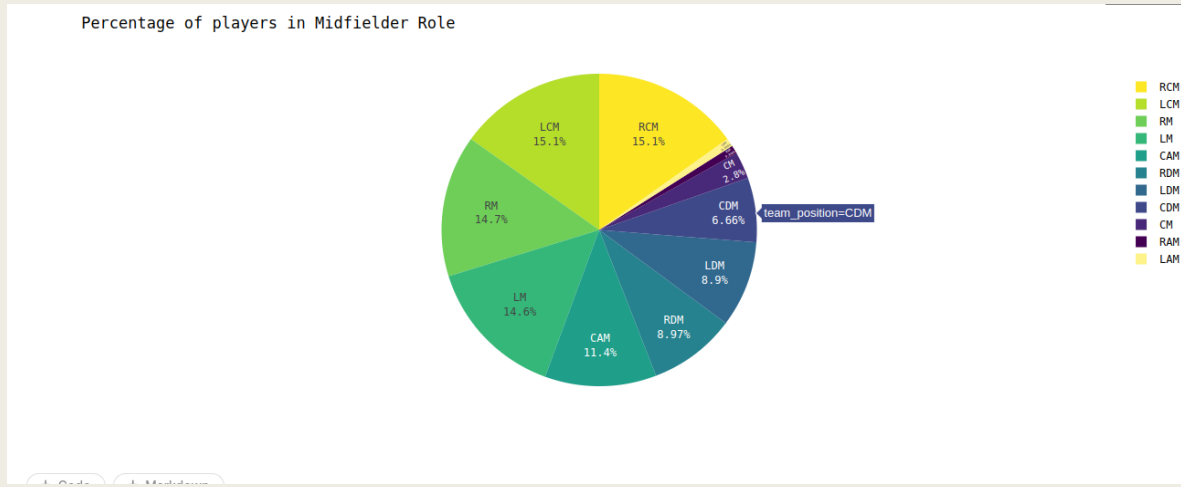



ID
Age
Overall
Potential
Special
International Reputation
Weak Foot
Skill Moves
Jersey Number
Height
Weight
Crossing
Finishing
HeadingAccuracy
ShortPassing
Volleys
Dribbling
Curve
FKAccuracy
LongPassing
BallControl
Acceleration
SprintSpeed
Agility
Reactions
Balance
ShotPower
Jumping
Stamina
Strength
LongShots
Aggression
Interceptions
Positioning
Vision
Penalties
Composure
Marking
StandingTackle
SlidingTackle
GKDiving
GKHandling
GKKicking
GKPositioning
GKReflexes
Best Overall Rating
Bmi
0.2
0.0
−0.2
−0.4
−0.6

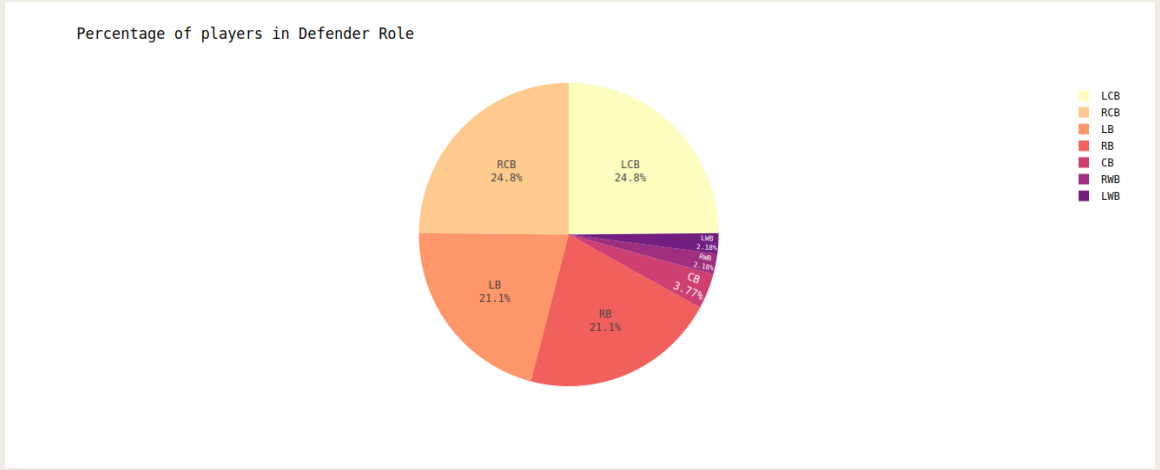
Dataset – EDA : Distribution of Data



Distribution of players in Attacking Role

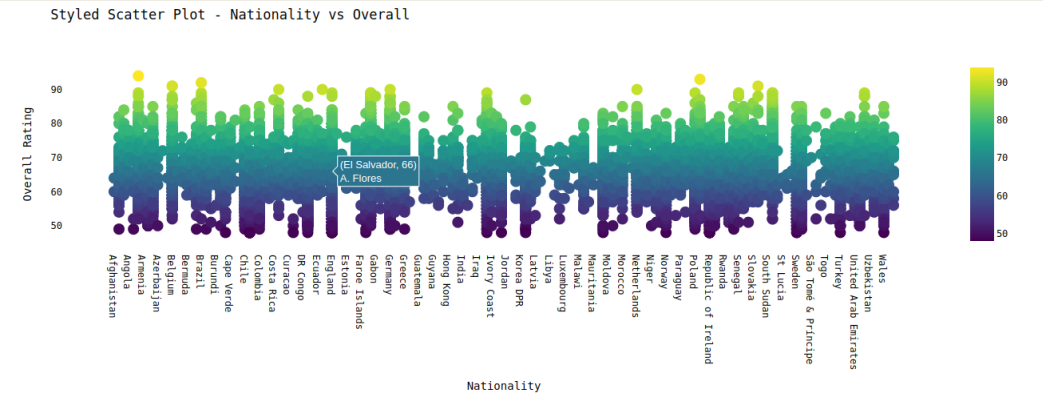


Distribution of players in Midfielder Role

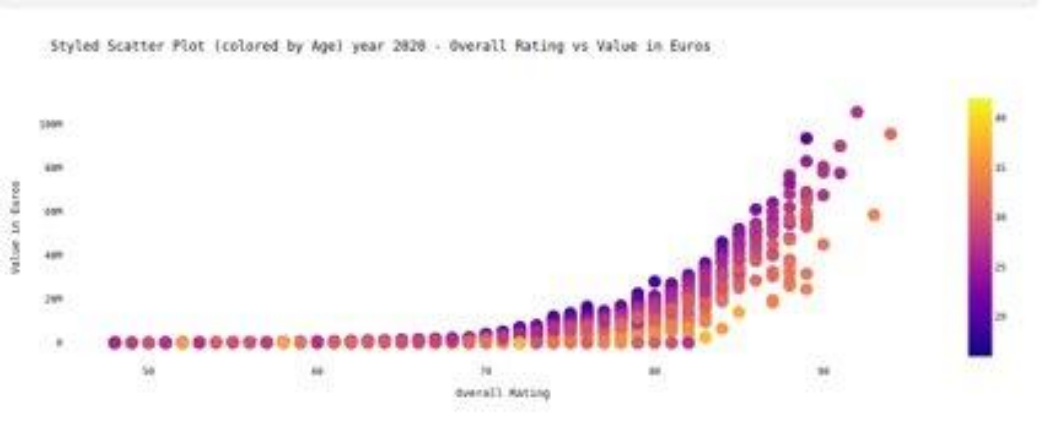


Distribution of players in Defender Role

Dataset – EDA : Interesting Observations



This scatter plot provides an overview of spread of player's rated across y-axis per nationality (x-axis). Also, this scatter plot highlights that specific countries with more players playing the game at league level hinting towards possible scouting destination. (Top recommended players in our models are from Europe, South America.)



This scatter plot illustrates the value (Millions) of a player based on their overall rating. Our focus is on top right corner. Notice that players with highest overall rating are valued more based on age grp 25-30. Players above 30 yrs are valued less despite high overall. (This pattern is realized in our models)

Dataset – Pre-Processing

Our Roadmap


K-Nearest-Neighbours
Bert
Linear Regression and Neural Networks
Club-Philosophy and Clustering
Modified Neural Networks and XGBoost Regression

Approach 1 : k-Nearest-Neighbours

Aim : Given a player, recommend k most similar players.

The value of a data point is determined by the data points around it.

Data: Removed all textual data and knn will be performed on all columns containing numeric data.

Working:

The input data is represented as a set of points in a multi-dimensional space, where each point represents an instance of the dataset.

When a player is given as an input, the algorithm searches for the k nearest neighbours to this new instance within the dataset.

The distance metric used to calculate the distance is Euclidian Distance.

Algorithm: KD Tree

Analysis

Pros:

- Simple and easy to interpret

Cons:

- Very slow as the number of data points increases because the model needs to store all data points.
- Not memory efficient
- Sensitive to outliers. Outliers also have a vote!
- The model recommends players which generally have similar ratings but it fails to take into account the team philosophy and the compatibility of a particular player with the club and team manager's style of play.

```
recommend_similar(176580) # Players similar to Luis Suarez
✓ 0.1s
```

```
Name:  R. Lewandowski
Overall: 91
Market Value: €€86M
Age: 30
BMI: 23.87

Name:  K. Benzema
Overall: 88
Market Value: €€53M
Age: 31
BMI: 23.62

Name:  Hulk
Overall: 80
Market Value: €€11.5M
Age: 32
BMI: 26.08

Name:  E. Cavani
Overall: 86
Market Value: €€35.5M
Age: 32
BMI: 22.43
```

Approach 2 : Transformers and NLP model

Aim : Given a club, rank players that match the club traits/tags

The value of a data point is determined by the data points around it.

Data: Used textual data which is "player_tags"

long_name	age	club	player_tags
Lionel Andrés Messi Cuccittini	32	FC Barcelona	#Dribbler, #Distance Shooter, #Crosser, #FK Specialist, #Acrobat, #Clinical Finisher, #Complete Forw...
Cristiano Ronaldo dos Santos Aveiro	34	Juventus	#Speedster, #Dribbler, #Distance Shooter, #Acrobat, #Clinical Finisher, #Complete Forward

Approach 2 : Transformers and NLP model

Working:

- BERT was used to create embeddings of player tags which were in string form
- All rows were grouped by 'club' and inserted into a new data frame with 'club' and 'embeddings' as feature
- Clubs were mapped to the mean of all player 'embeddings' - "Club Philosophy"

```
    return sbert_model.encode(sent)
```

```
[ ] new_df['embeddings'] = new_df['player_traits'].apply(constructEmbeddings)
    # new_df = pd.read_csv('/content/drive/MyDrive/College/Sem 6/RecSys/Project2/embeddings.csv')
```

```
[ ] new_df['embeddings']
```

```
0       [-0.36305606, 0.77589226, 0.55432, 0.27569258,...
1       [-0.48443338, 1.1557947, 0.91646373, 0.4993325...
2       [-0.6640025, 0.46728063, 0.8076286, 0.26393124...
3       [0.09925105, -0.5052245, 1.3611902, 0.6164345,...
4       [-0.49783152, 0.51528376, 1.1077756, 0.2626912...
                              ...
7561    [0.010219832, -0.21413945, 1.6238617, 0.744068...
7562    [-0.14079884, -0.10212083, 1.5475837, -0.04730...
7563    [0.0011217035, -0.4691528, 1.5869671, 0.630030...
7564    [-0.024325462, -0.46545637, 0.7181439, 0.51775...
```

```
yes = new_df.groupby('club')['embeddings'].mean()
```

```
yes
```

```
club
 SSV Jahn Regensburg      [-0.13267958, -0.02988186, 0.9670204,
1. FC Heidenheim 1846     [-0.06787688, -0.049798835, 1.2745366,
1. FC Kaiserslautern      [0.04257611, -0.23833944, 1.5059918, 0
1. FC Köln                [-0.08155326, -0.08018833, 1.1886828,
1. FC Magdeburg           [-0.13525926, 0.26306897, 1.3279678, 0
                                                    ...
Yokohama F. Marinos       [-0.18142563, -0.0682986, 1.0145396, 0
Zagłębie Lubin            [-0.1889395, 0.26326638, 0.8993827, 0.
Çaykur Rizespor           [-0.25040364, 0.016386658, 1.6851532,
Örebro SK                 [0.12261691, -0.025407264, 1.2110761,
Östersunds FK             [-0.14924082, -0.17977697, 1.3357896,
Name: embeddings, Length: 691, dtype: object
```

Approach 2 : Transformers and NLP model

Working:

- 'similarity_score' is calculated based on cosine similarity between the player embeddings and the group club embeddings
- This is done for all players and the list is sorted in descending order

```
[ ]  def applyCosine(player_vect, club_vect = yes['FC Barcelona']):
       return cosine(club_vect, player_vect)

     def getRankedList(clubName):
       club_vect = yes[clubName]
       player_df = new_df[['short_name','embeddings']]
       player_df['similarity_score'] = player_df['embeddings'].apply(applyCosine)
       return player_df

     out= getRankedList('FC Barcelona')
     out.describe
```

```
[ ]  out = out.sort_values(by='similarity_score', ascending=False)
     out.head(100)
     out[['short_name', 'similarity_score']].to_csv('output_FCB.csv')
```

short_name	similarity_score
Pozo	0.83840865
M. Pjaca	0.83840865
V. Kadlec	0.83840865
C. Musonda	0.83840865
A. Najar	0.83840865
S. Özkan	0.83840865
J. Cavallaro	0.83840865
J. Dayton	0.83840865
Gustavo Lobatelro	0.83840865
D. Mitchell	0.83840865
G. Mackay-Steven	0.83840865
J. Amoah	0.83840865
R. Kishna	0.83840865

Analysis (Transformers and NLP Model)

Pros

- Emphasis placed on an individual's playstyle
- Specific recommendations can be made which may help clubs find the perfect fit

Cons

- Dataset had very limited number (83) of tags
- Only 41% of the dataset had player_tags. They must be entered manually.
- A huge number of players ended up with the same similarity score due to a shortage of tags
- The model didn't take into account that players might be highly similar but not good.

Dataset – Feature Engineering (For Approach 3)

- This means that the extent to which a player improves (or fails to) depends not only on their own skills and characteristics, but also those of their teammates.
- The average overall rating of players at their club
- The average potential rating of players at their club

Approach 3

- **Aim** : To predict the improvement of a player when he transfers to a club.
- **Feature Engineering**: As mentioned in previous slide. The essential features are extracted using selectKBest.
- **Model**:
  - *Both a Neural network and a linear regression model where the target variable is overall rating change.*
  - *Metric for evaluation: MSE*
  - *Choose the 30 best players according to their improvement.*

Analysis

Linear regression : 0.81

Neural Network :

Optimizer	MSE
Adam	0.8
Adagrad	0.77
Adadelta	0.78

Epochs:100, Adagrad optimizer, batch size = 128

Learning rate	MSE
0.06	0.77
0.01	0.78
0.1	0.78

Shortcomings

- The models do not perform that well considering the range of values .
- We also see that the new club columns like average rating and potential do not affect the improvement attribute very much.
- Since they aren't that relevant , most clubs get the same recommendations...

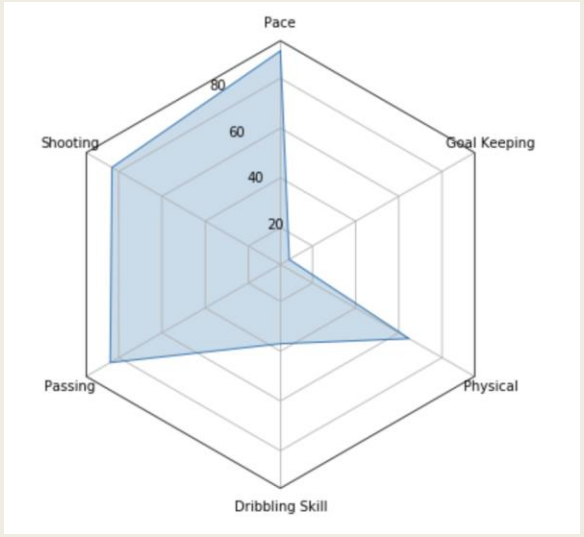
FINAL MODELS

Note: This clustering problem was not a straightforward neighborhood-based problem as we will see.

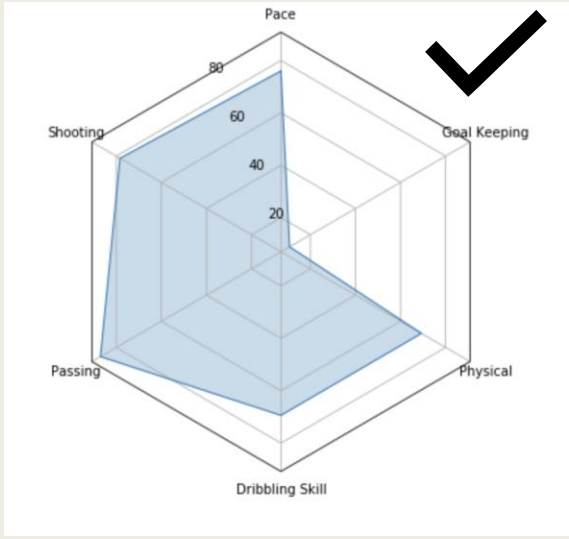
Approach 4 : Clustering

- We try to model **HOW** a player plays and not only how **GOOD** the player plays.
- Then match players and clubs which can play in the same way.
- Essentially, we recommend players which can play in the Club's system and are also good at what they do.

Club Blueprint



Player-1

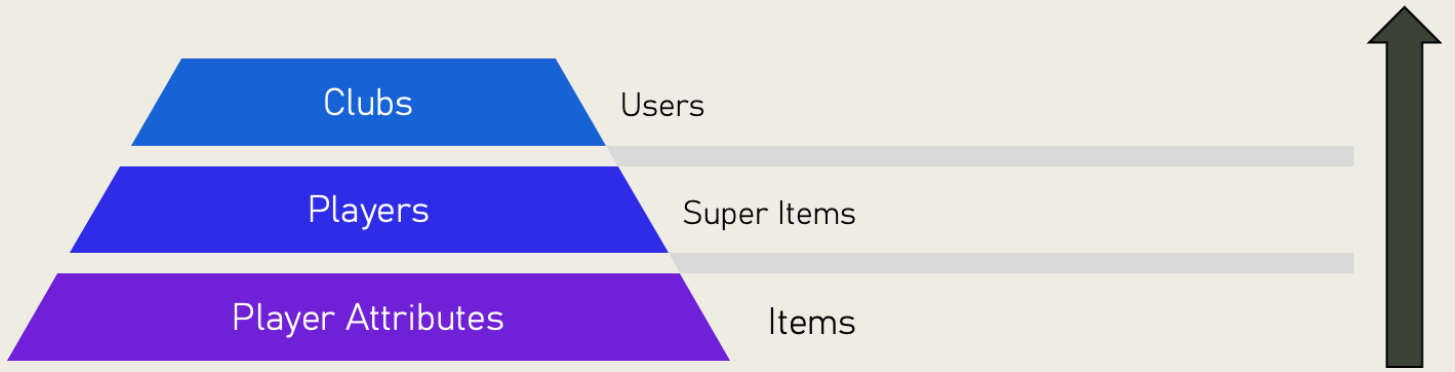


Player-2

Feature-Engineering

- We identified a 3-level hierarchy for our model, and asked the following questions:
  - What is a Position?
  - What is a Player?
  - What is a Club?

Problem is that User's rating for Super-Items do not exist.



Feature-Engineering

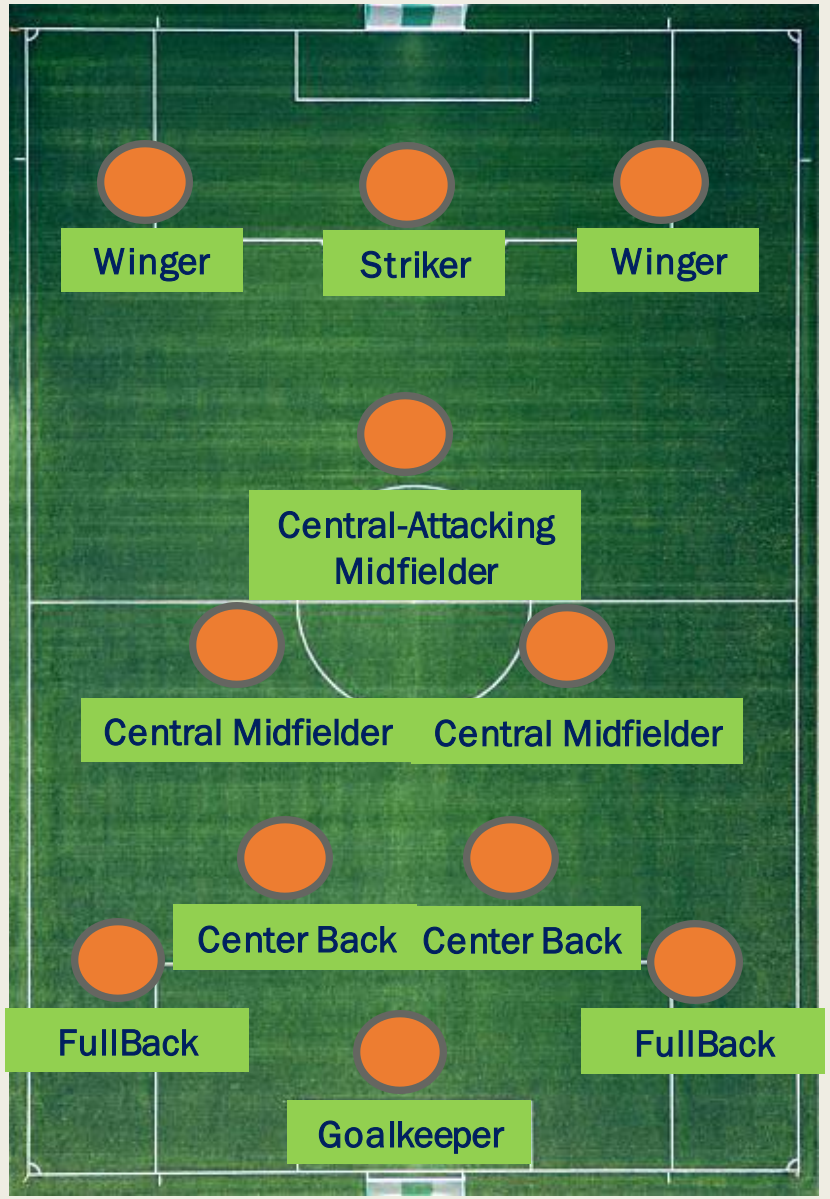
- **What is a Position?**
  - We divide the team/field into 7 different positions as is the convention.
  - But how do we define each position?
    - **Essential Attributes:** We take the top 10 attributes that affect the Overall rating of a player in the position.
    - How to find top 10? **Pearson's Correlation Coefficient**
    - Why top 10?

```
np.sort(fs.scores_)
✓ 0.0s
array([5.64069079e+00, 6.01175039e+01, 1.15560838e+02, 2.47649992e+02,
       3.93058587e+02, 4.06505762e+02, 9.81697813e+02, 1.11763132e+03,
       1.14061688e+03, 1.18550313e+03, 1.23847753e+03, 1.43281252e+03,
       1.45863958e+03, 2.17451588e+03, 2.75182023e+03, 2.87025967e+03,
       4.27010137e+03, 4.41289126e+03, 5.73828589e+03, 5.77270709e+03,
       6.38939382e+03, 6.72582876e+03, 6.82025961e+03, 7.05054983e+03,
       1.01421237e+04, 1.12446741e+04, 1.53094768e+04, 2.02789365e+04,
       2.03344285e+04, 2.53797729e+04, 2.61312201e+04, 3.10141541e+04,
       4.00066369e+04, 4.20135489e+04])
```

Center Backs

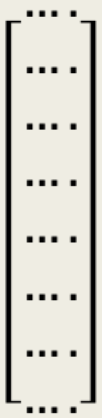
```
np.sort(fs.scores_)
✓ 0.0s
array([3.65345458e+01, 5.69809956e+01, 3.27125902e+02, 3.33799277e+02,
       3.94357894e+02, 4.06366012e+02, 4.20283099e+02, 5.74432383e+02,
       1.00840274e+03, 1.08777661e+03, 1.25534166e+03, 1.30754510e+03,
       1.31132367e+03, 1.48923981e+03, 1.65545309e+03, 1.82432936e+03,
       1.93064956e+03, 1.96564543e+03, 2.04720323e+03, 2.23125685e+03,
       3.29979350e+03, 4.40809608e+03, 4.78511333e+03, 5.21453946e+03,
       9.82067628e+03, 1.13220423e+04, 1.15921133e+04, 1.40356844e+04,
       1.65498676e+04, 2.95708181e+04, 3.14012974e+04, 3.48146197e+04,
       4.66352331e+04, 6.80187994e+04])
```

Striker

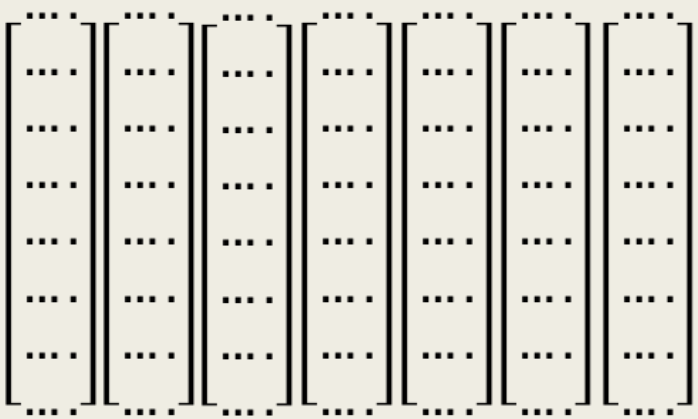


Feature-Engineering

- **What is a Player?**
  - Depending on the position of the player we represent the player according to the essential attributes of that position.

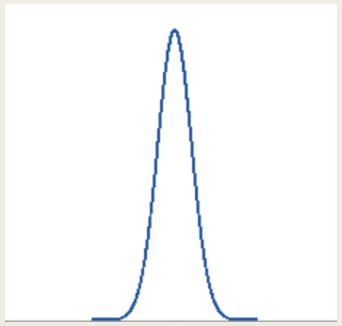


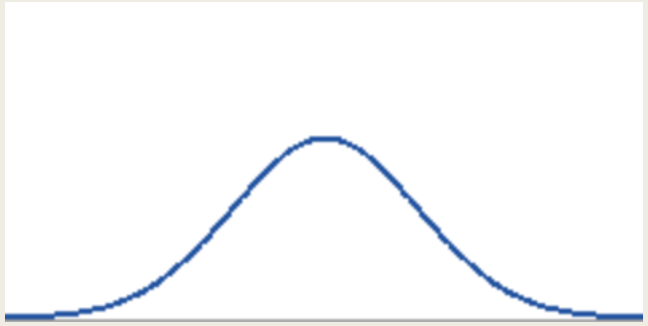
- **What is a Club?**
  - We consider clubs to be natural clusters of players.
  - Each club is made of 7 clusters, 1 each for the 7 distinct positions.
  - But how do we represent each positional cluster?
    - **Through the mean positional vector**
    - How to get the positional vector? Weighted mean of the club players in that position. (Over all years)
      - Higher weights given to players with high overall. (explicitly)
      - Higher weight given to players who stay longer at the club. (Implicitly)



Feature-Engineering

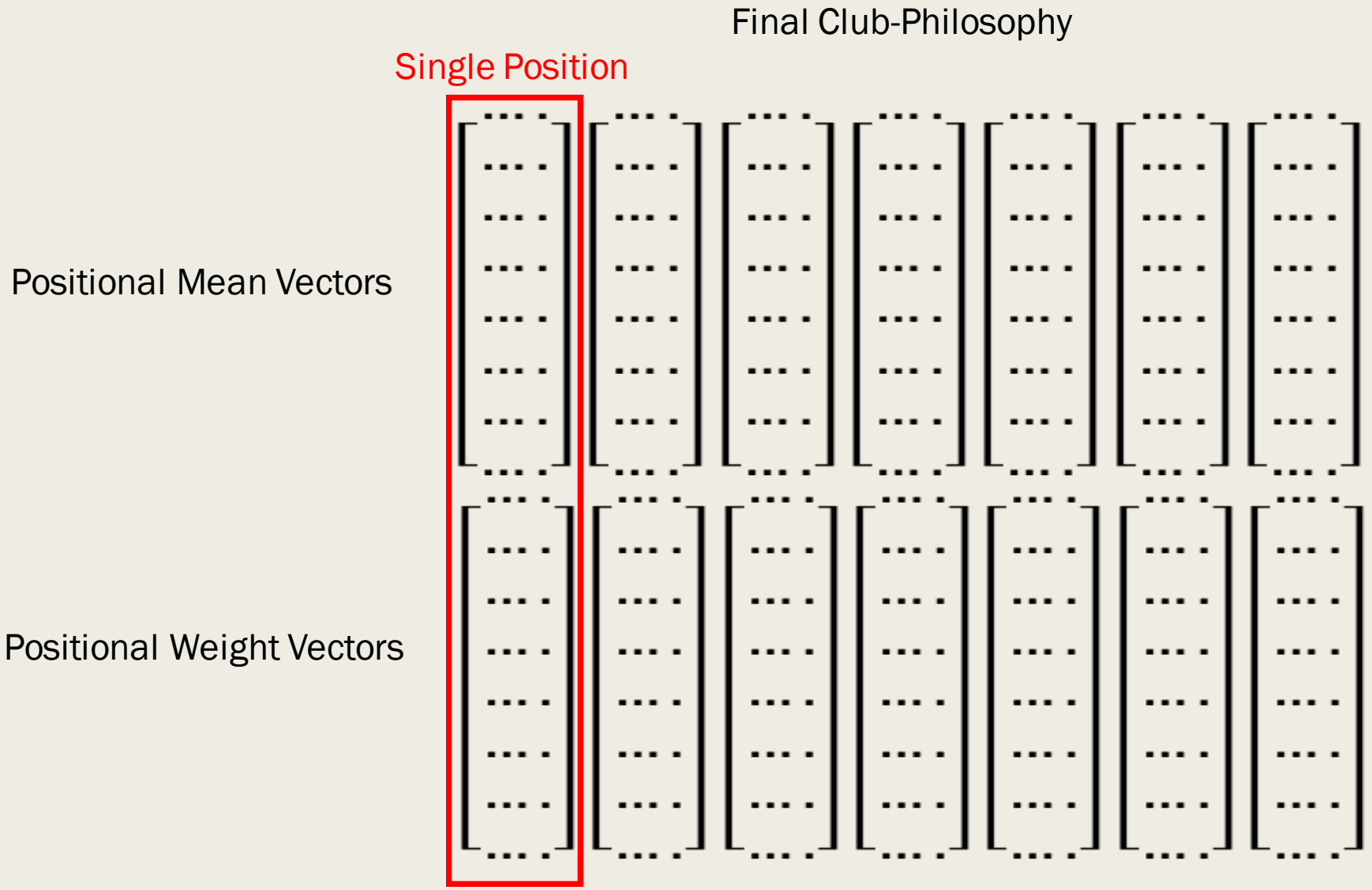
- Is the Club philosophy complete?
  - No, we have just defined how a club likes the positional attributes to be.
  - We still need to define how much the club likes them that way. (Standard Deviation)
    - Consider two clubs with "Pace" attribute of Centre-Back players distributed as shown:
    - Which Club prefers CBs to have certain pace?





  - We define weight for an attribute for a club position as 1/(Standard Deviation).
    - Taken over top 20 players in that position over all years in that club.
  - We take weights only for essential attributes of each position.

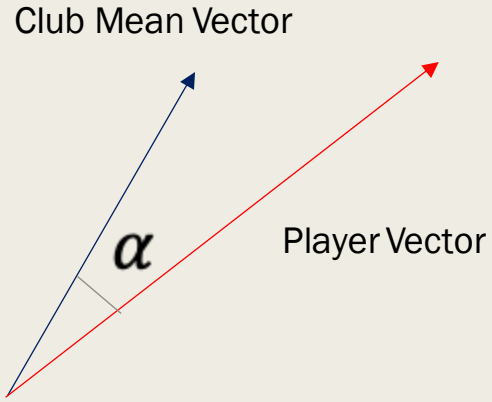
Feature-Engineering



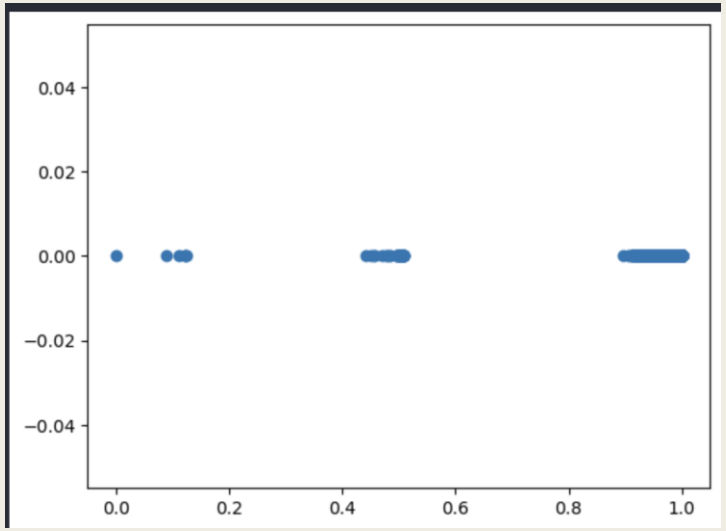
Weighted Pearson's Correlation Coefficient $$\frac{\Sigma\, w_i(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\Sigma(w_i(x_i-\bar{x})^2)}\,.\sqrt{\Sigma(w_i(y_i-\bar{y})^2)}}$$

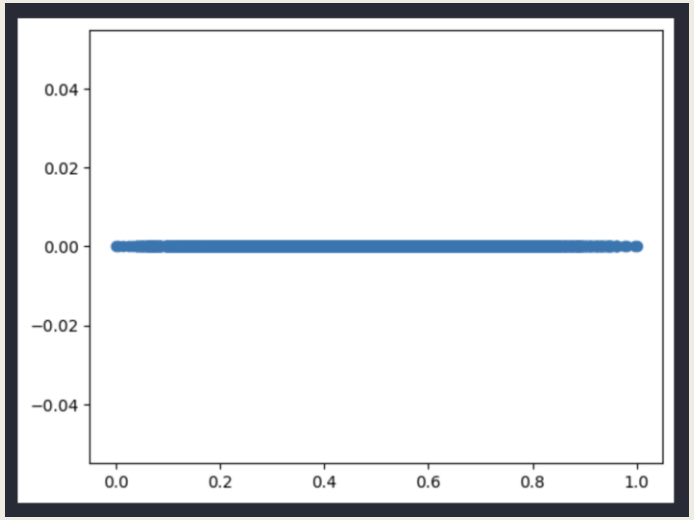
Working

- **How Similar the player is to Club's philosophy? Irrespective of how good or bad.**
  - In terms of Spider Plot: How similar the shape is, irrespective of area.
  - In terms of Positional Vector: How small angle α is, irrespective of vector magnitudes.
  - An obvious solution was Cosine Similarity, or in our case, weighted cosine similarity. But it did not show good results as it was not too discriminative.
  - Then we tried Weighted Pearson's Correlation Coefficient. But it was just making the players with high overalls similar.
    - To solve this we first normalized our vectors and then applied pearson's correlation coefficient. This did the job.



Weighted Cosine: Not Discriminative





Weighted PCC: Discriminative

Working

- Is sorting these similarities now enough?
  - Turns out it is not enough.
  - In our recommendations we get players who have high similarity but are very weak.
  - Their spider graphs have very small areas but shape is highly similar to what club wants.
  - The red boxes show the low strength measure of the player.

```
[[98781    D. Baumgartner
  Name: Name, dtype: object,
  229818,
  [1.0, 0.5640625350345524] ,
 [94676    J. O'Connell
  Name: Name, dtype: object,
  212300,
  [0.9963346855375808, 0.732298102137154]],
 [94722    K. Hause
  Name: Name, dtype: object,
  210635,
  [0.9601953058734722, 0.6630365865280212]],
 [99667    A. Miron
  Name: Name, dtype: object,
  248002,
  [0.9445985960407631, 0.4927540851340742]],
```

Working

- We introduced a strength measure in order to tackle this.
- Strength of a player for a club is the weighted magnitude of the player's vector.
  - The weights being the positional weights of the club

$$\sqrt{\frac{\Sigma(w_i(x_i)^2)}{\Sigma w_i}}$$

  - Now we Min-max scale the similarity ratings s_i and the weighted magnitudes m_i
  - The final rating is

$$r_i = \lambda_1 s_i + \lambda_2 m_i$$

We found the following values to be well balancing:

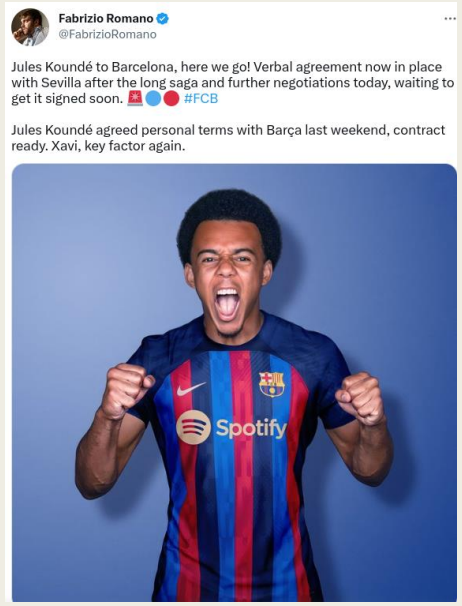
$$\lambda_1 = 0.63$$
$$\lambda_2 = 0.37$$

Comparisons with Real Life Patterns

- We queried recommendations in Centre-Back position for FC Barcelona
- Interesting resemblance with real life patterns was observed:

```
[[89061    M. Ginter
 Name: Name, dtype: object,
 207862,
 [0.9759302281700118, 0.8815359970174212]],
[88397    Éder Militão
 Name: Name, dtype: object,
 240130,
 [0.9443736657572159, 0.8360099497017297]],
[89840    J. Koundé
 Name: Name, dtype: object,
 241486,
 [0.9283980436334522, 0.8575857652502339]],
[89398    M. Akanji
 Name: Name, dtype: object,
 229237,
 [0.9548722052327286, 0.8011019133323763]],
[90733    J. Matip
 Name: Name, dtype: object,
 197061,
 [0.9092702490742487, 0.8771277788852602]],
[94661    A. Christensen
 Name: Name, dtype: object,
 213661,
 [0.9425134429495874, 0.8064332169161973]],
[89095    J. Stones
```

J. Kounde was picked by FC Barcelona at the start of 2023 season.



M. Akanji and J. Stones are players for Manchester City. There has been a rich history of FCB and MCI exchanging players.

Analysis (Pros and Cons)

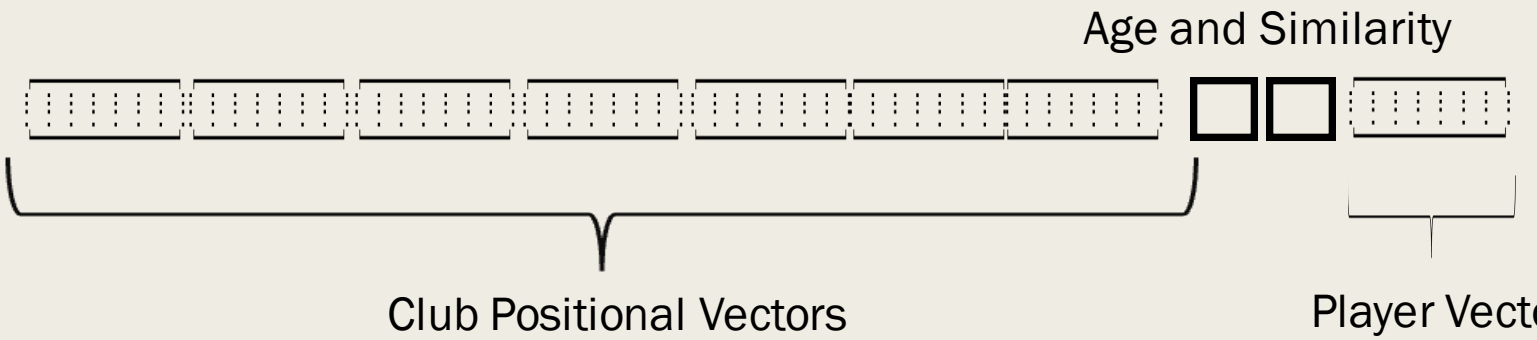
- Pros:
  - The model recommends players which have a very high chance of fitting into the club's system and thus performing well.
  - The model is dynamic in terms of age range and cost of transfer of the player that the club wants.
- Cons:
  - The model only recommends what the club thinks would work because it already works.
  - Is it necessary that players who are not as similar to club philosophy cannot fit as well or even better? Given how he connects with other players in the club(even in other positions.)
  - Lack of serendipity

Approach 5 : XGBoost and NN

- This model extends the idea of previous approach.
- We answer the question if its necessary that only the players similar to club's positional philosophy can fit in?
- Essentially, we recommend players based on how good they already are and how much they are predicted to improve if they join a given club.
- We take the club-philosophy, player vector and their similarity as the input features to a regression model.
- What is the target? The improvement in overall rating of the player when he joins a club.

Input Dataset Creation

- We treat every season as a new transfer for the player. (even if club remains same)
  - For ex. Transfer from: P-2019 of C1-2019 -> P-2020 of C2-2020
  - The improvement for this transfer is the training target for this transfer.
  - The training features are:
    - P-2019 Age
    - P-2019 positional vector
    - P-2019 similarity with C2-2019: For how much the player is already compatible
    - C2-2019 all positional mean vectors: For how much the player can improve when integrated with other players.



For Ex. A striker with great heading ability could score a lot if the midfielders have great crossing ability even though striker isn't very similar to club's striker mold

Neural Network: Analysis

Optimizer	Mean Squared Error
Adam	0.6826
Adagrad	0.6823
Adadelta	0.6840

Epochs:200, Adam optimizer

Learning rate	Mean Squared Error
0.001	0.6826
0.01	0.6853
0.1	0.6824

Improvement Values ranged from –3.5 to 7.5 after standard scaling.

XGBoost: Analysis

- XGBoost
  - Number of estimators = 1000
  - Depth = 6
  - Learning rate = 0.01
  - Mean Squared Error = 0.685

- XGBoost with Important Features
  - We did something interesting here:
    - XGBoost, after being trained can give out importance scores for the input features.
    - We noticed that Age and Similarity are important features in determining improvement.
    - Now, we set a threshold on the importance measure so that top-k features are filtered out.
    - Using these filtered feature set we trained another XGBoost model which had MSE = 0.67

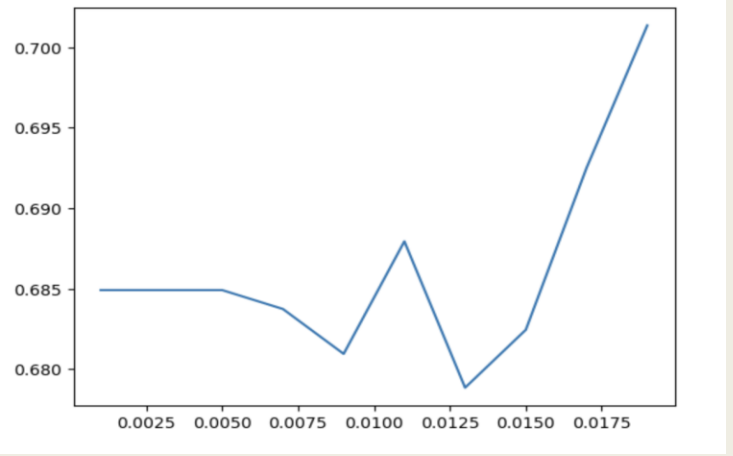
Age and Similarity have high importance scores

```
print(my_model.feature_importances_)
```

```
[0.03273191 0.00851815 0.00635185 0.0078481  0.0088491  0.01566508
 0.01592314 0.01828243 0.03617587 0.0132319  0.01224892 0.00798366
 0.00894529 0.00907722 0.00808141 0.01350628 0.01073165 0.01127564
 0.00801901 0.00924683 0.00880632 0.01266967 0.01039656 0.00947976
 0.01312917 0.00758138 0.00847808 0.01772967 0.01019354 0.02454421
 0.01179754 0.01323241 0.01150341 0.01590569 0.01068776 0.01739481
 0.01001501 0.01229732 0.01069494 0.00916138 0.00795054 0.0108042
 0.0118127  0.00856235 0.01347495 0.01145019 0.01121532 0.01107345
 0.01118613 0.00749227 0.01145793 0.0099155  0.00909001 0.01236769
 0.0111692  0.01094123 0.01500657 0.01052115 0.01233654 0.01015758
 0.01821454 0.01053189 0.01652169 0.00988168 0.01187515 0.01600802
 0.01362472 0.01186936 0.01428742 0.00905751 0.01449527 0.01531938
 0.0092067  0.01076354 0.0090439  0.01095743 0.01149364 0.01012931
 0.01012998 0.01159441 0.01350665 0.01511029]
```

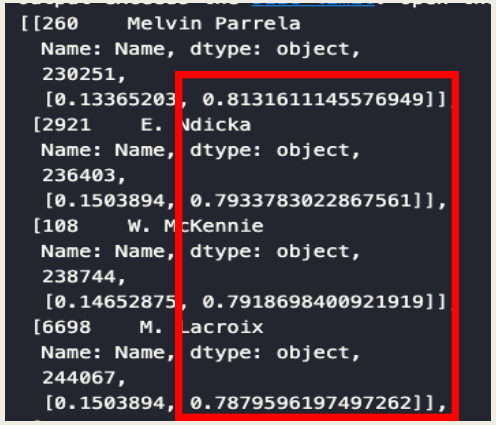
Determining the threshold = 0.013

```
import matplotlib.pyplot as plt
plt.plot(thresholds, y)
plt.show()
```



Comparisons with Real Life Patterns

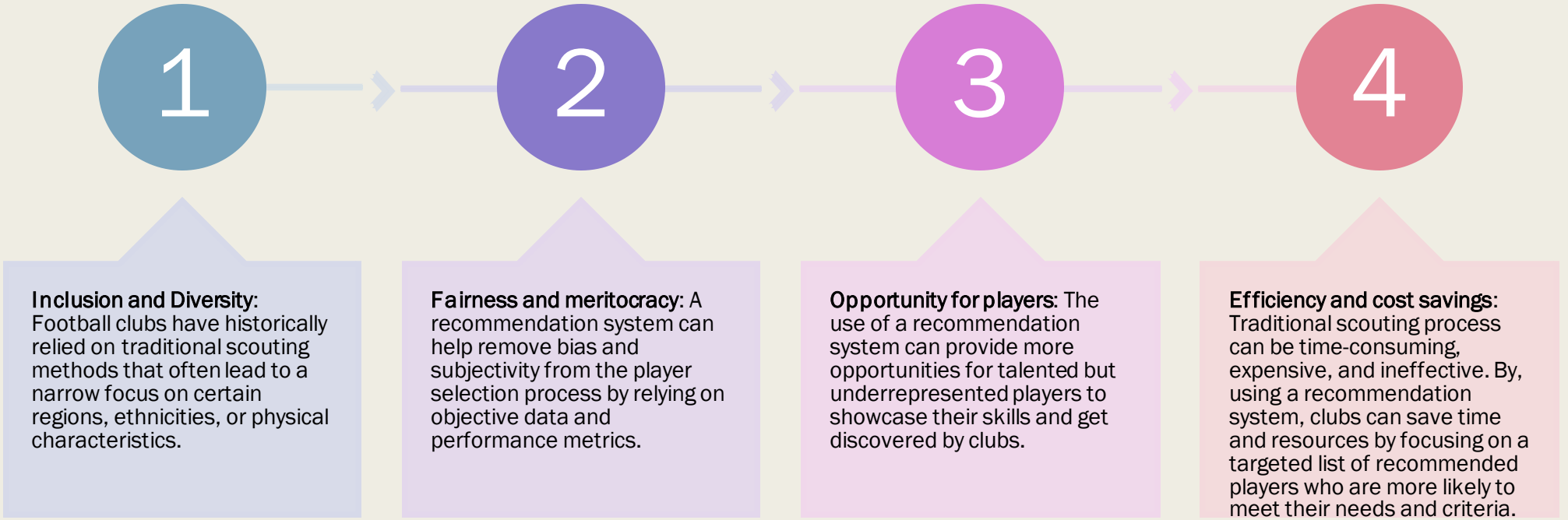
- Both NN and XGBoost models modeled the fact that players that are rated very high already, do not have much room for improvement.
  - For recommending, we again drew a balance between player strength and predicted improvement.
- Another pattern in real-life is that a highly rated player takes a dip in performance when he joins a new club.
  - Reason are lesser playing time initially and just time to fit in.
  - This fact was modeled by XGBoost and NN. Many high rated players had negative predicted improvements when joining a new club.



Players with strength 0.9 are not coming on top, modelling both patterns.

SUMMARY

Social Impact



AI-ML Ideas and Concepts

Recommendation paradigms: Collaborative filtering and content based filtering.

ML models: Linear regression , KNN , XGBoost

Deep Learning : Neural networks

NLP : Bert

Statistics: Mean and variance.

Data pre-processing

Feature engineering: Creation of new features from existing data .

Novelty

- Using 5 different models and analysing their advantages and shortcomings.
- A user friendly frontend that lets even the layman use the recommendation system.
- A sophisticated definition of club philosophy that completely describes the type of players in the club and the type of players that will fit well into the club (Data synthesis).
- Using Important feature selection to improve XGBoost perfomance
- Using an NLP model like Bert which uses the description of players to recommend players.

Given only player data we had to define

Dataset Creation

Future Scope

- Features in text form containing information about the players playstyle could help boost performance of NLP and transformer-based models.
- A "Money Ball" like system where we recommend high rated players who have been overlooked by other clubs. The club using the system will be able to get good layers at a lower price.
- A club recommendation for players, in case a player has multiple offers , using the improvement based model.
- Refine metrics and features.
- Since the FIFA data is sequential year-wise data, RNNs can be use.

THANK YOU