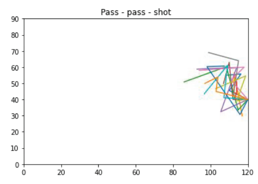
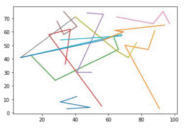
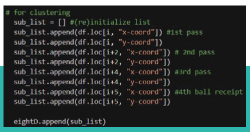


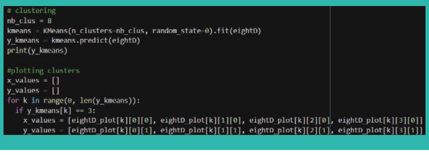
Project





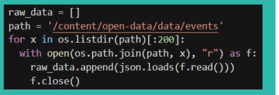
```
# for clustering
  sub_list = [] #(re)initialize list
  sub_list.append(df.loc[i, "x-coord"]) #1st pass
  sub_list.append(df.loc[i, "y-coord"])
  sub_list.append(df.loc[i+2, "x-coord"]) # 2nd pass
  sub_list.append(df.loc[i+2, "y-coord"])
  sub_list.append(df.loc[i+4, "x-coord"]) #3rd pass
  sub_list.append(df.loc[i+4, "y-coord"])
  sub_list.append(df.loc[i+5, "x-coord"]) #4th ball receipt
  sub_list.append(df.loc[i+5, "y-coord"])

  eightD.append(sub_list)
```



```
# clustering
nb_clus = 8
kmeans = KMeans(n_clusters=nb_clus, random_state=0).fit(eightD)
y_kmeans = kmeans.predict(eightD)
print(y_kmeans)

#plotting clusters
x_values = []
y_values = []
for k in range(0, len(y_kmeans)):
  if y_kmeans[k] == 3:
    x_values = [eightD_plot[k][0][0], eightD_plot[k][1][0], eightD_plot[k][2][0], eightD_plot[k][3][0]]
    y_values = [eightD_plot[k][0][1], eightD_plot[k][1][1], eightD_plot[k][2][1], eightD_plot[k][3][1]]
```



```
raw_data = []
path = '/content/open-data/data/events'
for x in os.listdir(path)[:200]:
  with open(os.path.join(path, x), "r") as f:
    raw_data.append(json.loads(f.read()))
    f.close()
```

Project

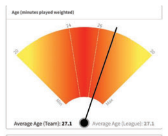
- In the paper: pass-pass-pass
- Our goal: pass-pass-goal

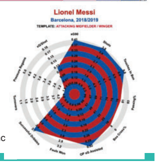
Workflow

1. Clean up data: from json file to pandas dataframe
2. Filter out unwanted events & only keep wanted information
3. Cluster event combination
4. Plot clustered event combinations

Data

- StatsBomb: football analytics
- On GitHub: 100s of games
  - https://github.com/statsbomb/open-data
  - 1 game = 1 json file
- 3600 (+/- 500) events / game
  - shot, pass, ball receipt, pressure, half start, carry, block, …
  - 43 different events





https://www.sportperformanceanalysis.com/article/statsbomb-advanced-football-analytics-through-an-interactive-visualisation-platform

Example: random event from a game

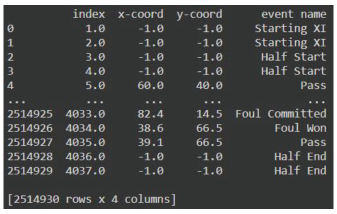
```
{'id': '7629f193-adba-4986-86b8-5e854e5eccd6', 'index': 17, 'period':
1, 'timestamp': '00:00:08.394', 'minute': 0, 'second': 8, 'type':
{'id': 30, 'name': 'Pass'}, 'possession': 2, 'possession_team':
{'id': 217, 'name': 'Barcelona'}, 'play_pattern': {'id': 9, 'name':
'From Kick Off'}, 'team': {'id': 217, 'name': 'Barcelona'}, 'player':
{'id': 5470, 'name': 'Ivan Rakitić'}, 'position': {'id': 13, 'name':
'Right Center Midfield'}, 'location': [50.0, 67.0], 'duration': 2.9,
'related_events': ['1236d54d-dfea-46fd-a421-2ce498c75b0d'], 'pass':
{'recipient': {'id': 5213, 'name': 'Gerard Piqué Bernabéu'},
'length': 27.45906, 'angle': -1.7539071, 'height': {'id': 1, 'name':
'Ground Pass'}, 'end_location': [45.0, 40.0], 'body_part': {'id': 40,
'name': 'Right Foot'}}}
```

Random event from a game

Id
Index
Period
Timestamp
Minute
Second
Type (id, name)
Possession
Possession team (id, name)
Play pattern (id, name)
Team (id, name)
Player (id, name)
Position (id, name)
Location (x, y)
Duration
Related events
Pass (recipient (id, name), length, angle, height (id, name), end location (x, y), body part (id, name))

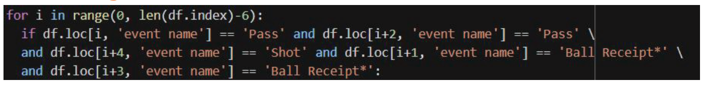
From json to data frame

- Extract useful information
- [-1, -1] as coordinate for events without location

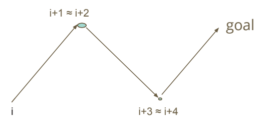

```
         index  x-coord  y-coord       event name
0          1.0     -1.0     -1.0      Starting XI
1          2.0     -1.0     -1.0      Starting XI
2          3.0     -1.0     -1.0       Half Start
3          4.0     -1.0     -1.0       Half Start
4          5.0     60.0     40.0             Pass
...        ...      ...      ...              ...
2514925  4033.0    82.4     14.5   Foul Committed
2514926  4034.0    38.6     66.5        Foul Won
2514927  4035.0    39.1     66.5             Pass
2514928  4036.0    -1.0     -1.0         Half End
2514929  4037.0    -1.0     -1.0         Half End

[2514930 rows x 4 columns]
```

Selecting events


```
for i in range(0, len(df.index)-6):
  if df.loc[i, 'event name'] == 'Pass' and df.loc[i+2, 'event name'] == 'Pass' \
  and df.loc[i+4, 'event name'] == 'Shot' and df.loc[i+1, 'event name'] == 'Ball Receipt*' \
  and df.loc[i+3, 'event name'] == 'Ball Receipt*':
```

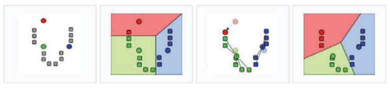
- **i = pass**
- i+1 = ball receipt
- **i+2 = pass**
- i+3 = ball receipt
- **i+4 = shot**

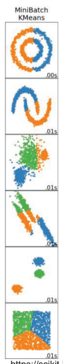


- Put coordinates of the events into a list
- Stack lists together

K-means clustering

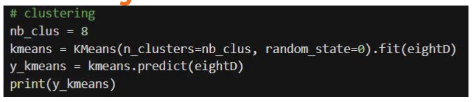
- Scikit-learn
- Implementation: (2N)-dimensional vector
  - 2 because of x- & y-coordinates
  - N = number of events
  - In our case: 4 events (pass - pass - shot - ball in goal)


https://en.wikipedia.org/wiki/K-means_clustering



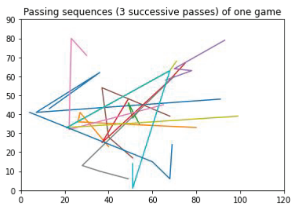
https://scikit-learn.org/stable/modules/clustering.html#clustering

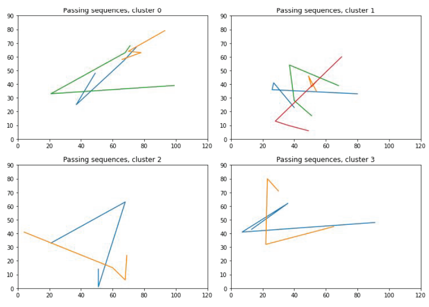
Clustering: code


```
# clustering
nb_clus = 8
kmeans = KMeans(n_clusters=nb_clus, random_state=0).fit(eightD)
y_kmeans = kmeans.predict(eightD)
print(y_kmeans)
```

- Input: eightD, an array
  - [[$x_{1,1}$, $y_{1,1}$, $x_{1,2}$, $y_{1,2}$, $x_{1,3}$, $y_{1,3}$, $x_{1,4}$, $y_{1,4}$] [$x_{2,1}$, $y_{2,1}$, $x_{2,2}$, $y_{2,2}$, $x_{2,3}$, $y_{2,3}$, $x_{2,4}$, $y_{2,4}$] [...] ...]
- nb_clus: sets number of clusters
- Output: y_kmeans, list, in which cluster each event combination is
  - [2 4 1 0 5 5 4 ... 7 3 1 1]

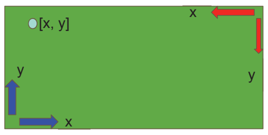
Example: 3 consecutive passses in a game, 4 clusters





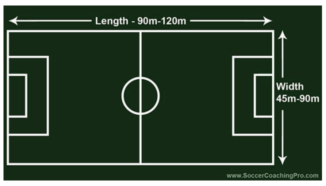
Note on location #1

- x- & y-coordinates relative to the team that has ball possession
  - Red team has ball: red coordinates [x, y] = [100, 10]
  - Blue team has ball: blue coordinates [x, y] = [10, 90]



Note on location #2

- Football field dimensions aren't always the same
- FIFA recommendation: 105m * 68m
- Goal size fixed: 7.3m in width



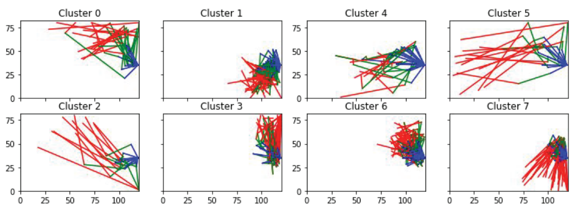
Problems and solutions

- Problem 1
  - Shot: location from where ball was shot in team A's coordinate system
  - Shot + 1 event: location in team B's coordinates
- Solution 1: transform coordinates

BUT

- Problem 2
  - field dimensions not the same everywhere
  - goal not always at the same coordinates
- Solution 2: Assume fixed goal coordinates -> error relatively small

Pass - pass - shot clustering

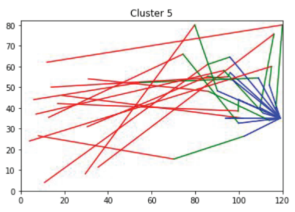


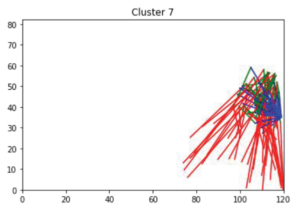
Red: 1st pass

Green: 2nd pass

Blue: shot

Pass - pass - shot clustering





Conclusions & take home messages

- With very little, we can do a lot
- Limitations & how to improve:
  - RAM & running time
  - Goal coordinates
  - Better graphs (see paper)
- Use:
  - How do different teams play?
  - Analyse own & opponents playing strategy