master_fpl

September 23, 2022

[1]:
```
import pandas as pd
import numpy as np
import warnings
import matplotlib.pyplot as plt
import seaborn as sns

pd.set_option('display.max_columns', None)
warnings.filterwarnings('ignore')
```

[2]:
```
master_df = pd.read_csv("master_fpl_data.csv")

drop_useless = ['Unnamed: 0', 'transfers_balance','transfers_in',␣
 ↪'transfers_out','team_a_score', \
            'team_h_score','round', 'team_x' , 'kickoff_time', 'opponent_team']

master_df = master_df.drop(drop_useless, axis=1)

master_df.columns
```

[2]:
```
Index(['season_x', 'name', 'position', 'assists', 'bonus', 'bps',
       'clean_sheets', 'creativity', 'element', 'fixture', 'goals_conceded',
       'goals_scored', 'ict_index', 'influence', 'minutes', 'opp_team_name',
       'own_goals', 'penalties_missed', 'penalties_saved', 'red_cards',
       'saves', 'selected', 'threat', 'total_points', 'value', 'was_home',
       'yellow_cards', 'GW'],
      dtype='object')
```

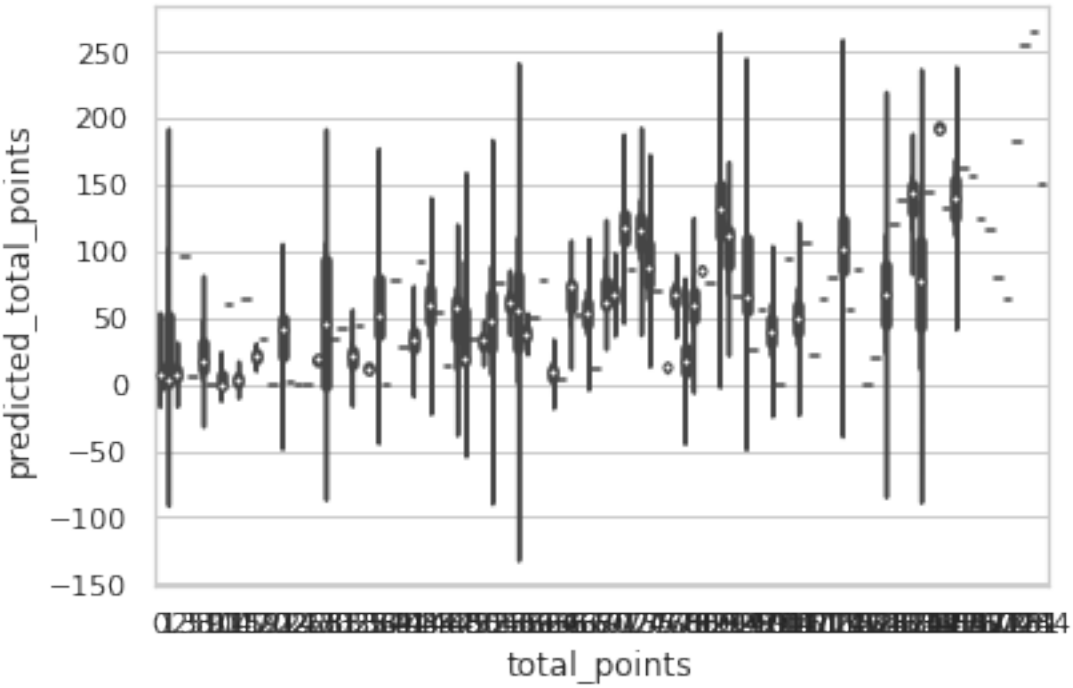
[5]:
```
X_axis = "total_points"

filtered = "predicted_total_points"

more_than = 100

n_season = "2020-21"

n1_season = "2021-22"
```

[10]:
```
attribute_cols = ['name','creativity', 'ict_index', 'influence',\
                  'threat', 'total_points']
attribute_df = master_df[(master_df.season_x == n_season)][attribute_cols].
 ↪groupby(['name']).sum()

attribute_df.reset_index(inplace=True)
# attribute_df.sort_values("total_points", ascending=False).head()
info_cols = ['name', 'position','selected', 'value']
if_df = master_df[(master_df.GW == 1)& (master_df.season_x ==␣
 ↪n1_season)][info_cols]
if_df.reset_index(inplace=True, drop=True)
X = if_df.merge(attribute_df, "inner", on="name")
print(X.columns)

Y = master_df[(master_df.season_x == n1_season)][["name",'total_points']].
 ↪groupby(['name']).sum()
Y.reset_index(inplace=True)
Y.rename(columns={"total_points": "predicted_total_points"}, inplace=True)
# Y.sort_values("predicted_total_points", ascending=False).head()
print(Y.columns)

XY = Y.merge(X[X.position == "MID"], "inner", on="name")

print(XY.columns)
print(XY.shape)

XY = XY[['name', 'total_points', 'position', 'selected', 'value',\
 'creativity', 'ict_index', 'influence', 'threat','predicted_total_points']]
# XY.sort_values("influence", ascending=False).head()
```

```
Index(['name', 'position', 'selected', 'value', 'creativity', 'ict_index',
       'influence', 'threat', 'total_points'],
      dtype='object')
Index(['name', 'predicted_total_points'], dtype='object')
Index(['name', 'predicted_total_points', 'position', 'selected', 'value',
       'creativity', 'ict_index', 'influence', 'threat', 'total_points'],
      dtype='object')
(178, 10)
```

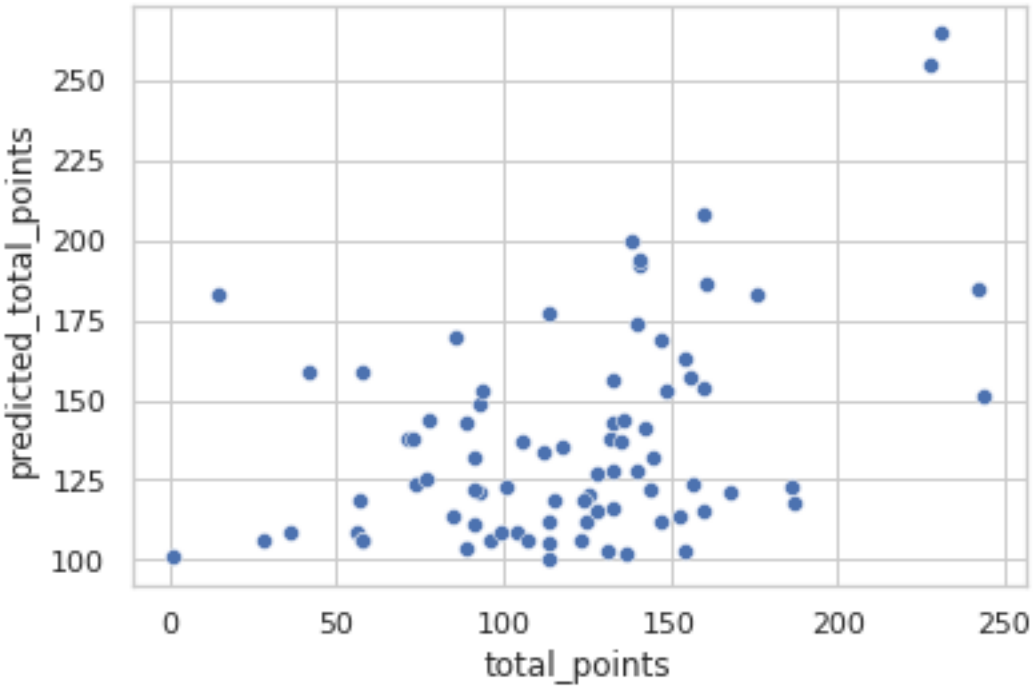
[11]:
```
sns.set(style = 'whitegrid')

sns.violinplot(x =X_axis,\
               y ="predicted_total_points",\
               data = XY[[X_axis, "predicted_total_points"]])
```

[11]:
```
<AxesSubplot:xlabel='total_points', ylabel='predicted_total_points'>
```



[318]:
```
sns.scatterplot(x =X_axis,\
                y ="predicted_total_points",\
                data = XY[[X_axis, "predicted_total_points"]])
```

[318]: <AxesSubplot:xlabel='total_points', ylabel='predicted_total_points'>



[36]:
```
def correlation(master):

    list_corr = []
    attribute_cols = ['name','creativity', 'ict_index', 'influence','threat',␣
↪'total_points']
    info_cols = ['name', 'position','selected', 'value']

    no_seasons = len(master.season_x.unique())
    seasons = master.season_x.unique().tolist()

    for i in range(no_seasons-1):
        attribute_df = master[(master.season_x == seasons[i])][attribute_cols].
↪groupby(['name']).sum()
        attribute_df.reset_index(inplace=True)

        if_df = master[(master.GW == 1)& (master.season_x ==␣
↪seasons[i+1])][info_cols]
        if_df.reset_index(inplace=True, drop=True)

        X = if_df.merge(attribute_df, "inner", on="name")
        Y = master[(master.season_x == seasons[i+1])][["name",'total_points']].
↪groupby(['name']).sum()
        Y.reset_index(inplace=True)
```

```
        Y.rename(columns={"total_points": "predicted_total_points"},␣
↪inplace=True)


        XY = Y.merge(X[(X.position == "MID") & (X.total_points >= 100)],␣
↪"inner", on="name")


        list_corr.append(XY.corr()["predicted_total_points"].tolist()[::-1])


    return list_corr


    pass
```

[37]:
```
list_corr = correlation(master_df)

corr = pd.DataFrame(columns=['total_points', 'selected', 'value', 'creativity',␣
↪'ict_index',
       'influence', 'threat', 'predicted_total_points'])


for i in list_corr:
    corr.loc[len(corr)] = i

corr.loc[len(corr)] = corr.sum().tolist()


corr.loc[len(corr)-1]/5
```

[37]:
```
total_points             0.554182
selected                 0.597009
value                    0.429730
creativity               0.514702
ict_index                0.126206
influence                0.609509
threat                   0.506193
predicted_total_points   1.000000
Name: 5, dtype: float64
```

[47]:
```
TA_df = pd.read_csv("TransferAlgorithm.csv", encoding='latin-1')
TA_df.columns
```

[47]:
```
Index(['No.', ' BCV ', 'Position', 'Player', 'Team', ' Price ',
       ' Weighted minutes ', ' Weighted UPPM ', ' PPG - longer term ',
       'Fixture ratio', '1', '2', '3', '4', '5', '6', '7', 'Unnamed: 17'],
```

```
      dtype='object')
```

[51]:
```
TA_df = TA_df[[' BCV ', 'Position', 'Player', 'Team', ' Price ',\
       ' Weighted minutes ', ' Weighted UPPM ', ' PPG - longer term ',\
       'Fixture ratio', '1', '2', '3', '4', '5', '6', '7',]]

TA_df
```

[51]:

	BCV	Position	Player	Team	Price	Weighted minutes	\
0	0.26	M	Fabio Vieira	ARS	6.0	31.00	
1	0.51	M	Saka	ARS	8.0	77.00	
2	0.29	M	Smith-Rowe	ARS	6.0	39.00	
3	0.19	M	Elneny	ARS	4.5	15.00	
4	0.19	M	Sambi Lokonga	ARS	4.5	8.00	
..	…	…	…	…	…	…	
394	0.23	GK	Gazzaniga	FUL	4.0	NaN	
395	0.47	GK	Guaita	CPL	4.5	NaN	
396	0.42	GK	McCarthy	SOT	4.5	NaN	
397	0.07	GK	Bazunu	SOT	4.5	NaN	
398	0.01	GK	Johnstone	CPL	4.5	NaN	

	Weighted UPPM	PPG - longer term	Fixture ratio	1	2	3	\
0	0.061	1.89	106%	1.78	2.10	2.04	
1	0.057	4.37	106%	4.10	4.85	4.72	
2	0.054	2.10	106%	1.97	2.33	2.27	
3	0.031	0.47	106%	0.44	0.52	0.51	
4	0.032	0.25	106%	0.24	0.28	0.27	
..	…	…	…	…	…	…	
394	NaN	NaN	NaN	1.08	1.48	1.52	
395	NaN	NaN	NaN	2.86	2.39	2.90	
396	NaN	NaN	NaN	2.44	2.66	2.38	
397	NaN	NaN	NaN	0.61	0.87	0.81	
398	NaN	NaN	NaN	0.43	0.47	0.57	

	4	5	6	7
0	2.27	2.06	1.76	2.13
1	5.23	4.75	4.07	4.92
2	2.51	2.28	1.96	2.37
3	0.56	0.51	0.44	0.53
4	0.30	0.28	0.24	0.29
..	…	…	…	…
394	1.37	1.57	1.37	1.42
395	2.33	2.87	2.50	2.29
396	2.42	2.37	2.39	2.29
397	0.84	0.76	0.85	0.91
398	0.50	0.67	0.61	0.60

```
[399 rows x 16 columns]
```

[321]:
```
epl16_df = master_df[master_df["season_x"]=='2016-17']
epl17_df = master_df[master_df["season_x"]=='2017-18']
epl18_df = master_df[master_df["season_x"]=='2018-19']
epl19_df = master_df[master_df["season_x"]=='2019-20']
epl20_df = master_df[master_df["season_x"]=='2020-21']
epl21_df = master_df[master_df["season_x"]=='2021-22']
```

[322]:
```
epl16_df = master_df[master_df["season_x"]=='2016-17']
epl17_df = master_df[master_df["season_x"]=='2017-18']
epl18_df = master_df[master_df["season_x"]=='2018-19']
epl19_df = master_df[master_df["season_x"]=='2019-20']
epl20_df = master_df[master_df["season_x"]=='2020-21']
epl21_df = master_df[master_df["season_x"]=='2021-22']

cleaned_epl16_df = epl16_df.groupby(["name","position"],␣
 ↪as_index=False)[['goals','assists', 'minutes','clean_sheets','ict',␣
 ↪'points', 'value']].sum()
cleaned_epl16_df = cleaned_epl16_df.drop(["value"], axis=1).merge(epl16_df.
 ↪iloc[::-1].drop_duplicates(subset=["name"])[["name", "value" ]], on="name")
cleaned_epl16_df["cost"] = cleaned_epl16_df.value
cleaned_epl16_df = cleaned_epl16_df.drop(["value"], axis=1)
cleaned_epl16_df["value"] = cleaned_epl16_df["points"]/
 ↪cleaned_epl16_df["cost"]*10
cleaned_epl16_df.value = cleaned_epl16_df.value.astype("int")
cleaned_epl16_df.cost = cleaned_epl16_df.cost.astype("int")/10
cleaned_epl16_df["matches"] = cleaned_epl16_df.minutes/90
cleaned_epl16_df.matches =  cleaned_epl16_df.matches.apply(lambda x:round(x,2))
cleaned_epl16_df["point_per_match"] = cleaned_epl16_df.points/cleaned_epl16_df.
 ↪matches
cleaned_epl16_df.point_per_match =  cleaned_epl16_df.point_per_match.
 ↪apply(lambda x:round(x,2))


cleaned_epl17_df = epl17_df.groupby(["name","position"],␣
 ↪as_index=False)[['goals','assists', 'minutes','clean_sheets','ict',␣
 ↪'points', 'value']].sum()
cleaned_epl17_df = cleaned_epl17_df.drop(["value"], axis=1).merge(epl17_df.
 ↪iloc[::-1].drop_duplicates(subset=["name"])[["name", "value" ]], on="name")
cleaned_epl17_df["cost"] = cleaned_epl17_df.value
cleaned_epl17_df = cleaned_epl17_df.drop(["value"], axis=1)
cleaned_epl17_df["value"] = cleaned_epl17_df["points"]/
 ↪cleaned_epl17_df["cost"]*10
cleaned_epl17_df.value = cleaned_epl17_df.value.astype("int")
cleaned_epl17_df.cost = cleaned_epl17_df.cost.astype("int")/10
```

```
cleaned_epl17_df["matches"] = cleaned_epl17_df.minutes/90
cleaned_epl17_df.matches =  cleaned_epl17_df.matches.apply(lambda x:round(x,2))
cleaned_epl17_df["point_per_match"] = cleaned_epl17_df.points/cleaned_epl17_df.
 ↪matches
cleaned_epl17_df.point_per_match =  cleaned_epl17_df.point_per_match.
 ↪apply(lambda x:round(x,2))


cleaned_epl18_df = epl18_df.groupby(["name","position"], as_index=False)[[␣
 ↪'goals','assists', 'minutes','clean_sheets','ict', 'points', 'value']].sum()
cleaned_epl18_df = cleaned_epl18_df.drop(["value"], axis=1).merge(epl18_df.
 ↪iloc[::-1].drop_duplicates(subset=["name"])[["name", "value" ]], on="name")
cleaned_epl18_df["cost"] = cleaned_epl18_df.value
cleaned_epl18_df = cleaned_epl18_df.drop(["value"], axis=1)
cleaned_epl18_df["value"] = cleaned_epl18_df["points"]/
 ↪cleaned_epl18_df["cost"]*10
cleaned_epl18_df.value = cleaned_epl18_df.value.astype("int")
cleaned_epl18_df.cost = cleaned_epl18_df.cost.astype("int")/10
cleaned_epl18_df["matches"] = cleaned_epl18_df.minutes/90
cleaned_epl18_df.matches =  cleaned_epl18_df.matches.apply(lambda x:round(x,2))
cleaned_epl18_df["point_per_match"] = cleaned_epl18_df.points/cleaned_epl18_df.
 ↪matches
cleaned_epl18_df.point_per_match =  cleaned_epl18_df.point_per_match.
 ↪apply(lambda x:round(x,2))


cleaned_epl19_df = epl19_df.groupby(["name","position"], as_index=False)[[␣
 ↪'goals','assists', 'minutes','clean_sheets','ict', 'points', 'value']].sum()
cleaned_epl19_df = cleaned_epl19_df.drop(["value"], axis=1).merge(epl19_df.
 ↪iloc[::-1].drop_duplicates(subset=["name"])[["name", "value" ]], on="name")
cleaned_epl19_df["cost"] = cleaned_epl19_df.value
cleaned_epl19_df = cleaned_epl19_df.drop(["value"], axis=1)
cleaned_epl19_df["value"] = cleaned_epl19_df["points"]/
 ↪cleaned_epl19_df["cost"]*10
cleaned_epl19_df.value = cleaned_epl19_df.value.astype("int")
cleaned_epl19_df.cost = cleaned_epl19_df.cost.astype("int")/10
cleaned_epl19_df["matches"] = cleaned_epl19_df.minutes/90
cleaned_epl19_df.matches =  cleaned_epl19_df.matches.apply(lambda x:round(x,2))
cleaned_epl19_df["point_per_match"] = cleaned_epl19_df.points/cleaned_epl19_df.
 ↪matches
cleaned_epl19_df.point_per_match =  cleaned_epl19_df.point_per_match.
 ↪apply(lambda x:round(x,2))


cleaned_epl20_df = epl20_df.groupby(["name","position"], as_index=False)[[␣
 ↪'goals','assists', 'minutes','clean_sheets','ict', 'points', 'value']].sum()
```

```
cleaned_epl20_df = cleaned_epl20_df.drop(["value"], axis=1).merge(epl20_df.
 ↪iloc[::-1].drop_duplicates(subset=["name"])[["name", "value" ]], on="name")
cleaned_epl20_df["cost"] = cleaned_epl20_df.value
cleaned_epl20_df = cleaned_epl20_df.drop(["value"], axis=1)
cleaned_epl20_df["value"] = cleaned_epl20_df["points"]/
 ↪cleaned_epl20_df["cost"]*10
cleaned_epl20_df.value = cleaned_epl20_df.value.astype("int")
cleaned_epl20_df.cost = cleaned_epl20_df.cost.astype("int")/10
cleaned_epl20_df["matches"] = cleaned_epl20_df.minutes/90
cleaned_epl20_df.matches =  cleaned_epl20_df.matches.apply(lambda x:round(x,2))
cleaned_epl20_df["point_per_match"] = cleaned_epl20_df.points/cleaned_epl20_df.
 ↪matches
cleaned_epl20_df.point_per_match =  cleaned_epl20_df.point_per_match.
 ↪apply(lambda x:round(x,2))


cleaned_epl21_df = epl21_df.groupby(["name","position"], as_index=False)[[␣
 ↪'goals','assists', 'minutes','clean_sheets','ict', 'points', 'value']].sum()
cleaned_epl21_df = cleaned_epl21_df.drop(["value"], axis=1).merge(epl21_df.
 ↪iloc[::-1].drop_duplicates(subset=["name"])[["name", "value" ]], on="name")
cleaned_epl21_df["cost"] = cleaned_epl21_df.value
cleaned_epl21_df = cleaned_epl21_df.drop(["value"], axis=1)
cleaned_epl21_df["value"] = cleaned_epl21_df["points"]/
 ↪cleaned_epl21_df["cost"]*10
cleaned_epl21_df.value = cleaned_epl21_df.value.astype("int")
cleaned_epl21_df.cost = cleaned_epl21_df.cost.astype("int")/10
cleaned_epl21_df["matches"] = cleaned_epl21_df.minutes/90
cleaned_epl21_df.matches =  cleaned_epl21_df.matches.apply(lambda x:round(x,2))
cleaned_epl21_df["point_per_match"] = cleaned_epl21_df.points/cleaned_epl21_df.
 ↪matches
cleaned_epl21_df.point_per_match =  cleaned_epl21_df.point_per_match.
 ↪apply(lambda x:round(x,2))
```

[323]:
```
cleaned_epl16_df[cleaned_epl16_df.points>=150].sort_values("point_per_match",␣
 ↪ascending=False).head(10)
```

[323]:

	name	position	goals	assists	minutes	clean_sheets	ict	\
86	Harry Kane	FWD	29	7	2523	14	336.0	
91	Heung-Min Son	MID	14	9	2063	10	255.6	
30	Bamidele Alli	MID	18	11	3036	17	301.7	
207	Sergio Agüero	FWD	20	5	2404	8	323.9	
200	Sadio Mané	MID	13	7	2242	8	246.0	
132	Kevin De Bruyne	MID	6	21	2877	11	336.6	
45	Christian Eriksen	MID	8	20	3159	16	389.7	
195	Romelu Lukaku	FWD	25	6	3266	13	354.1	
146	Marcos Alonso	DEF	6	5	2693	15	157.6	

```
127        Joshua King      FWD     16        3     2714             9  231.6

     points  cost  value  matches  point_per_match
86      224  10.7     20    28.03             7.99
91      174   7.7     22    22.92             7.59
30      225   8.2     27    33.73             6.67
207     175  13.1     13    26.71             6.55
200     156   9.0     17    24.91             6.26
132     199  10.7     18    31.97             6.22
45      218   8.2     26    35.10             6.21
195     221   9.6     23    36.29             6.09
146     177   5.9     30    29.92             5.92
127     178   5.4     32    30.16             5.90
```

[324]:
```
cleaned_epl16_df[cleaned_epl16_df.points >=140].
 ↪groupby(["position"])["point_per_match", "minutes"]\
.agg(["mean"]).sort_values(("point_per_match","mean"))
```

[324]:
```
          point_per_match      minutes
                     mean         mean
position
GK                4.270000  3077.000000
DEF               4.995000  3027.500000
MID               5.627273  2813.363636
FWD               6.165000  2795.000000
```

[325]:
```
cleaned_epl17_df[(cleaned_epl17_df.points>=160)].sort_values("point_per_match",␣
 ↪ascending=False).head(15)
#cleaned_epl17_df[(cleaned_epl17_df.points>=100) & (cleaned_epl17_df.position␣
 ↪== "DEF")].sort_values("value", ascending=False).head(30)
```

[325]:
```
                 name position  goals  assists  minutes  clean_sheets    ict  \
211      Mohamed Salah      MID     32       12     2905            15  454.4
242    Raheem Sterling      MID     18       17     2584            13  329.7
268      Sergio Agüero      FWD     21        6     1960            13  302.5
119      Heung-Min Son      MID     12        8     2292            14  249.1
115         Harry Kane      FWD     29        2     3074            15  407.4
171    Kevin De Bruyne      MID      8       18     3073            19  368.9
246       Riyad Mahrez      MID     12       13     2948             9  282.6
252    Roberto Firmino      FWD     15        8     2760            16  312.7
55   Christian Eriksen      MID     11       10     3218            17  358.2
34       Bamidele Alli      MID      9       13     2957            16  258.8
190      Marcos Alonso      DEF      7        2     2855            13  207.1
254      Romelu Lukaku      FWD     16        7     2866            14  273.2
139        Jamie Vardy      FWD     20        2     3248             8  248.6
231        Pascal Groß      MID      7        8     2928            10  257.6
66   César Azpilicueta      DEF      2        6     3330            15  167.5
```

	points	cost	value	matches	point_per_match
211	303	9.2	32	32.28	9.39
242	229	8.1	28	28.71	7.98
268	169	11.5	14	21.78	7.76
119	178	7.9	22	25.47	6.99
115	217	12.8	16	34.16	6.35
171	209	10.0	20	34.14	6.12
246	195	8.4	23	32.76	5.95
252	181	8.5	21	30.67	5.90
55	199	9.7	20	35.76	5.56
34	175	9.4	18	32.86	5.33
190	165	7.0	23	31.72	5.20
254	162	11.8	13	31.84	5.09
139	183	8.6	21	36.09	5.07
231	164	5.7	28	32.53	5.04
66	175	6.6	26	37.00	4.73

[326]:
```
cleaned_epl17_df[cleaned_epl17_df.points >=140].
 ↪groupby(["position"])["point_per_match", "minutes"]\
.agg(["mean"]).sort_values(("point_per_match","mean"))
```

[326]:

	point_per_match	minutes
	mean	mean
position		
GK	4.182857	3305.571429
DEF	5.050000	2868.500000
FWD	6.034000	2781.600000
MID	6.109091	2837.454545

[353]:
```
cleaned_epl18_df[(cleaned_epl18_df.points>=160)].sort_values("points",␣
 ↪ascending=False).head(25)

#cleaned_epl18_df[(cleaned_epl18_df.points>=150) & (cleaned_epl18_df.position␣
 ↪== "DEF")].sort_values("value", ascending=False).head(30)
```

[353]:

	name	position	goals	assists	minutes	\
261	Mohamed Salah	MID	22	12	3254	
291	Raheem Sterling	MID	17	15	2770	
315	Sadio Mané	MID	22	3	3076	
23	Andrew Robertson	DEF	0	12	3216	
351	Virgil van Dijk	DEF	4	4	3384	
289	Pierre-Emerick Aubameyang	MID	22	7	2722	
325	Sergio Agüero	FWD	21	11	2457	
346	Trent Alexander-Arnold	DEF	1	13	2460	
132	Gylfi Sigurdsson	MID	13	6	3122	
310	Ryan Fraser	MID	7	14	3165	

292	Raúl Jiménez	FWD	13	10	3112
283	Paul Pogba	MID	13	10	3007
39	Aymeric Laporte	DEF	3	3	3056
19	Alisson Ramses Becker	GK	0	0	3420
160	Jamie Vardy	FWD	18	5	2726
103	Ederson Santana de Moraes	GK	0	1	3420
58	Callum Wilson	FWD	14	12	2528
16	Alexandre Lacazette	FWD	13	12	2497
221	Luka Milivojevic	MID	12	2	3420
90	David Luiz Moreira Marinho	DEF	3	2	3238
229	Marcos Alonso	DEF	2	8	2759
184	Jordan Pickford	GK	0	0	3420
68	Christian Eriksen	MID	8	12	2770
302	Roberto Firmino	FWD	12	8	2606
134	Harry Kane	FWD	17	6	2423

	clean_sheets	ict	points	cost	value	matches	point_per_match
261	21	432.7	259	13.2	19	36.16	7.16
291	17	357.1	234	11.7	20	30.78	7.60
315	19	304.0	231	10.3	22	34.18	6.76
23	21	183.6	213	6.7	31	35.73	5.96
351	20	156.1	208	6.7	31	37.60	5.53
289	7	289.6	205	10.8	18	30.24	6.78
325	17	315.4	201	12.0	16	27.30	7.36
346	16	184.7	185	5.8	31	27.33	6.77
132	15	314.2	182	7.4	24	34.69	5.25
310	9	301.2	181	6.4	28	35.17	5.15
292	12	285.4	181	6.9	26	34.58	5.23
283	8	316.9	179	8.6	20	33.41	5.36
39	18	131.2	177	6.3	28	33.96	5.21
19	21	67.6	176	5.9	29	38.00	4.63
160	11	244.6	174	9.0	19	30.29	5.74
103	20	56.8	169	5.7	29	38.00	4.45
58	9	225.0	168	6.9	24	28.09	5.98
16	6	260.6	166	9.4	17	27.74	5.98
221	12	246.3	166	6.5	25	38.00	4.37
90	16	158.0	164	5.8	28	35.98	4.56
229	14	161.7	161	6.2	25	30.66	5.25
184	14	78.5	161	4.9	32	38.00	4.24
68	8	283.0	161	9.2	17	30.78	5.23
302	19	267.8	160	9.3	17	28.96	5.52
134	9	261.3	160	12.6	12	26.92	5.94

[328]:
```
cleaned_epl18_df[cleaned_epl18_df.points >=140].
 ↪groupby(["position"])["point_per_match", "minutes"]\
.agg(["mean"]).sort_values(("point_per_match","mean"))
```

[328]:

position	point_per_match mean	minutes mean
GK	4.235000	3315.000000
DEF	4.950000	3059.583333
MID	5.623571	2941.857143
FWD	5.632000	2657.500000

[352]:
```
cleaned_epl19_df[cleaned_epl19_df.points>=140].sort_values("points",
↪ascending=False).head(25)

#cleaned_epl19_df[(cleaned_epl19_df.points>=150)& (cleaned_epl19_df.position ==
↪"DEF")].sort_values("value", ascending=False).head(30)
```

[352]:

	name	position	goals	assists	minutes	clean_sheets	ict	points	cost	value	matches	point_per_match
282	Kevin De Bruyne	MID	13	23	2790	18	432.1	251	10.6	23	31.00	8.10
353	Mohamed Salah	MID	19	10	2879	16	387.4	233	12.5	18	31.99	7.28
428	Sadio Mané	MID	18	10	2745	12	292.8	221	12.1	18	30.50	7.25
223	Jamie Vardy	FWD	23	7	3032	11	288.8	210	9.7	21	33.69	6.23
471	Trent Alexander-Arnold	DEF	4	15	3173	14	275.3	210	7.7	27	35.26	5.96
396	Pierre-Emerick Aubameyang	MID	22	5	3136							
398	Raheem Sterling	MID	20	6	2651							
40	Anthony Martial	FWD	17	9	2625							
117	Danny Ings	FWD	22	2	2800							
399	Raúl Jiménez	FWD	17	7	3241							
29	Andrew Robertson	DEF	2	12	3111							
477	Virgil van Dijk	DEF	5	2	3420							
318	Marcus Rashford	MID	17	8	2645							
405	Riyad Mahrez	MID	11	14	1934							
370	Nick Pope	GK	0	0	3420							
195	Heung-Min Son	MID	11	13	2478							
487	Willian Borges Da Silva	MID	9	9	2595							
331	Matt Doherty	DEF	4	8	2836							
404	Richarlison de Andrade	FWD	13	5	3070							
130	Dean Henderson	GK	0	1	3240							
186	Harry Kane	FWD	18	2	2588							
274	Kasper Schmeichel	GK	0	0	3420							
411	Roberto Firmino	FWD	9	9	2985							
455	Tammy Abraham	FWD	15	6	2206							
419	Rui Pedro dos Santos Patrício	GK	0	0	3420							

396	10	282.4	205	10.9	18	34.84	5.88
398	13	350.2	204	12.0	17	29.46	6.92
40	14	259.3	200	8.5	23	29.17	6.86
117	6	302.3	198	7.6	26	31.11	6.36
399	15	325.5	194	8.0	24	36.01	5.39
29	13	210.3	181	7.0	25	34.57	5.24
477	15	151.7	178	6.5	27	38.00	4.68
318	9	271.3	177	8.9	19	29.39	6.02
405	10	279.8	175	8.6	20	21.49	8.14
370	15	93.1	170	5.2	32	38.00	4.47
195	7	254.6	169	9.7	17	27.53	6.14
487	10	294.8	168	7.3	23	28.83	5.83
331	12	160.0	167	6.5	25	31.51	5.30
404	9	252.2	165	8.2	20	34.11	4.84
130	13	70.7	160	5.2	30	36.00	4.44
186	6	230.6	158	11.0	14	28.76	5.49
274	13	89.6	156	5.5	28	38.00	4.11
411	16	301.0	155	9.3	16	33.17	4.67
455	7	223.3	153	7.3	20	24.51	6.24
419	13	74.2	153	5.3	28	38.00	4.03

[330]:
```
cleaned_epl19_df[cleaned_epl19_df.points >=140].
 ↪groupby(["position"])["point_per_match", "minutes"]\
.agg(["mean"]).sort_values(("point_per_match","mean"))
```

[330]:

	point_per_match	minutes
	mean	mean
position		
GK	4.095000	3390.000000
DEF	4.642857	3246.428571
FWD	5.843333	2729.444444
MID	6.454545	2685.272727

[350]:
```
cleaned_epl20_df[cleaned_epl20_df.points>=140].sort_values("points",␣
 ↪ascending=False).head(15)

#cleaned_epl20_df[(cleaned_epl20_df.points>=150)& (cleaned_epl20_df.position ==␣
 ↪"DEF")].sort_values("value", ascending=False).head(30)
```

[350]:

	name	position	goals	assists	minutes	\
94	Bruno Miguel Borges Fernandes	MID	18	14	3101	
247	Harry Kane	FWD	23	14	3083	
491	Mohamed Salah	MID	22	6	3077	
256	Heung-Min Son	MID	17	11	3119	
545	Patrick Bamford	FWD	17	11	3052	
306	Jamie Vardy	FWD	15	14	2838	
190	Emiliano Martínez	GK	0	0	3420	

```
603                    Sadio Mané      MID     11       11     2808
440                Marcus Rashford      MID     11       11     2917
635                  Stuart Dallas      DEF      8        3     3410
534                  Ollie Watkins      FWD     14        9     3328
174          Dominic Calvert-Lewin      FWD     16        6     2871
41                 Andrew Robertson      DEF      1        8     3384
275                  Jack Harrison      MID      8       10     2846
671         Trent Alexander-Arnold      DEF      2        8     3029

     clean_sheets    ict  points  cost  value  matches  point_per_match
94             13  396.2     244  11.3     21    34.46             7.08
247            12  355.9     242  11.9     20    34.26             7.06
491            11  385.8     231  12.9     17    34.19             6.76
256            13  315.2     228   9.6     23    34.66             6.58
545            10  274.6     194   6.6     29    33.91             5.72
306             8  248.3     187  10.2     18    31.53             5.93
190            15  108.5     186   5.3     35    38.00             4.89
603            12  315.5     176  11.8     14    31.20             5.64
440            12  260.3     174   9.6     18    32.41             5.37
635            12  174.2     171   5.5     31    37.89             4.51
534            14  314.0     168   6.3     26    36.98             4.54
174            11  249.0     165   7.5     22    31.90             5.17
41             12  224.5     161   7.3     22    37.60             4.28
275            11  225.2     160   5.6     28    31.62             5.06
671            10  255.7     160   7.8     20    33.66             4.75
```

[332]:
```
cleaned_epl20_df[cleaned_epl20_df.points >=140].
 ↪groupby(["position"])["point_per_match", "minutes"]\
.agg(["mean"]).sort_values(("point_per_match","mean"))
```

[332]:
```
          point_per_match      minutes
                      mean         mean
position
GK                4.345000  3144.375000
DEF               4.436667  3149.333333
FWD               5.481667  3001.500000
MID               5.658667  2748.266667
```

[349]:
```
cleaned_epl21_df[cleaned_epl21_df.points>=140].sort_values("points",␣
 ↪ascending=False).head(55)

#cleaned_epl21_df[(cleaned_epl21_df.points>=150)& (cleaned_epl21_df.position ==␣
 ↪"DEF")].sort_values("value", ascending=False).head(30)
```

[349]:
```
                                name position  goals  assists  \
506                  Mohamed Salah      MID     23       14
261                  Heung-Min Son      MID     23       10
```

690	Trent Alexander-Arnold	DEF	2	12	
369	João Pedro Cavaco Cancelo	DEF	1	11	
393	Kevin De Bruyne	MID	15	8	
317	Jarrod Bowen	MID	10	17	
44	Andrew Robertson	DEF	3	11	
249	Harry Kane	FWD	16	11	
703	Virgil van Dijk	DEF	3	3	
623	Sadio Mané	MID	16	5	
103	Bukayo Saka	MID	11	9	
33	Alisson Ramses Becker	GK	0	1	
176	Diogo Jota	MID	15	6	
469	Mason Mount	MID	11	11	
581	Raheem Sterling	MID	13	7	
143	Cristiano Ronaldo dos Santos Aveiro	FWD	18	3	
68	Aymeric Laporte	DEF	4	2	
335	Joel Matip	DEF	2	3	
306	James Ward-Prowse	MID	10	5	
297	James Maddison	MID	10	9	
189	Ederson Santana de Moraes	GK	0	0	
263	Hugo Lloris	GK	0	0	
89	Bernardo Mota Veiga de Carvalho e Silva	MID	8	8	
100	Bruno Miguel Borges Fernandes	MID	10	7	
55	Antonio Rüdiger	DEF	3	3	
367	José Malheiro de Sá	GK	0	1	
238	Gabriel Magalhães	DEF	5	0	
712	Wilfried Zaha	MID	14	1	
480	Matthew Cash	DEF	4	3	
582	Raphael Dias Belloli	MID	11	4	
621	Rúben Santos Gato Alves Dias	DEF	2	5	

	minutes	clean_sheets	ict	points	cost	value	matches	\
506	2758	17	417.0	265	13.1	20	30.64	
261	2919	17	327.7	255	11.2	22	32.43	
690	2853	18	268.1	208	8.4	24	31.70	
369	3137	19	227.8	200	7.2	27	34.86	
393	2106	13	284.2	194	12.1	16	23.40	
317	2897	8	251.0	192	6.9	27	32.19	
44	2537	17	187.4	186	7.3	25	28.19	
249	3141	13	316.1	185	12.5	14	34.90	
703	3060	21	133.1	183	6.8	26	34.00	
623	2818	17	288.7	183	11.8	15	31.31	
103	2888	13	305.9	177	6.7	26	32.09	
33	3150	20	68.8	174	6.1	28	35.00	
176	2267	14	250.0	170	8.3	20	25.19	
469	2358	11	265.8	169	7.7	21	26.20	
581	2121	13	248.4	163	10.5	15	23.57	
143	2454	8	274.6	159	12.2	13	27.27	

68	2740	18	112.3	159	6.0	26	30.44
335	2700	17	125.8	159	5.3	30	30.00
306	3125	7	246.4	157	6.4	24	34.72
297	2274	5	196.4	156	6.9	22	25.27
189	3240	20	55.2	154	6.2	24	36.00
263	3330	15	75.6	153	5.5	27	37.00
89	2766	16	245.8	153	7.0	21	30.73
100	3110	8	303.3	151	11.6	13	34.56
55	2945	15	128.9	149	6.1	24	32.72
367	3285	11	96.7	146	5.2	28	36.50
238	2991	13	127.5	144	5.3	27	33.23
712	2579	11	224.7	144	6.9	20	28.66
480	3197	13	154.3	143	5.3	26	35.52
582	2826	5	240.7	143	6.3	22	31.40
621	2401	14	107.6	141	6.2	22	26.68

	point_per_match
506	8.65
261	7.86
690	6.56
369	5.74
393	8.29
317	5.96
44	6.60
249	5.30
703	5.38
623	5.84
103	5.52
33	4.97
176	6.75
469	6.45
581	6.92
143	5.83
68	5.22
335	5.30
306	4.52
297	6.17
189	4.28
263	4.14
89	4.98
100	4.37
55	4.55
367	4.00
238	4.33
712	5.02
480	4.03
582	4.55

```
621             5.28
```

[334]:
```
cleaned_epl21_df.columns
```

[334]:
```
Index(['name', 'position', 'goals', 'assists', 'minutes', 'clean_sheets',
       'ict', 'points', 'cost', 'value', 'matches', 'point_per_match'],
      dtype='object')
```

[335]:
```
cleaned_epl21_df[cleaned_epl21_df.points >=140].
 ↪groupby(["position"])["point_per_match", "minutes"]\
.agg(["mean"]).sort_values(("point_per_match","mean"))
```

[335]:

	point_per_match	minutes
	mean	mean
position		
GK	4.347500	3251.250000
DEF	5.299000	2856.100000
FWD	5.565000	2797.500000
MID	6.123333	2654.133333

[336]:
```
cost_df = pd.read_csv("cost_data.csv")
```

[342]:
```
t17 = cleaned_epl17_df[(cleaned_epl17_df.points>=50)].
 ↪sort_values("point_per_match", ascending=False).reset_index(drop=True)
t18 = cleaned_epl18_df[(cleaned_epl18_df.points>=50)].
 ↪sort_values("point_per_match", ascending=False).reset_index(drop=True)
t19 = cleaned_epl19_df[(cleaned_epl19_df.points>=50)].
 ↪sort_values("point_per_match", ascending=False).reset_index(drop=True)
t20 = cleaned_epl20_df[(cleaned_epl20_df.points>=50)].
 ↪sort_values("point_per_match", ascending=False).reset_index(drop=True)
t21 = cleaned_epl21_df[(cleaned_epl21_df.points>=50)].
 ↪sort_values("point_per_match", ascending=False).reset_index(drop=True)

# t17["rank"] = t17.index *0.8
# t18["rank"] = t18.index *0.9
# t19["rank"] = t19.index *0.95
# t20["rank"] = t20.index *1.0
# t21["rank"] = t21.index *1.0

t_all = pd.DataFrame()
# t_all = t_all.append(t17)
# t_all = t_all.append(t18)
# t_all = t_all.append(t19)
# t_all = t_all.append(t20)
t_all = t_all.append(t21)
```

```
clean_t = t_all.groupby(["name", "position",'goals', 'assists', 'minutes',␣
 ↪'clean_sheets'])\
["point_per_match", "points"]\
.agg({"point_per_match":["mean"], "points": "mean"})\
.sort_values(("point_per_match", "mean"), ascending=False)

clean_t.reset_index(inplace=True)
clean_t[("point_per_match", "mean")] = clean_t[("point_per_match", "mean")].
 ↪apply(lambda x:round(x,2))
clean_t[("points", "mean")] = clean_t[("points", "mean")].apply(lambda x:
 ↪round(x,2))
#clean_t.sort_values(("point_per_match", "mean"), ascending=False)
# clean_t[clean_t["count"] > 1]
# clean_t["avg"] = clean_t["sum"]/clean_t["count"]
# clean_t["avg"] = clean_t.avg.apply(lambda x:round(x,2))
# clean_t = clean_t.sort_values("sum",ascending=False)
# clean_t.sort_values("avg", ascending=False)
```

[345]:
```
df_all.columns
```

[345]:
```
Index([                      'name',                  'now_cost',
                  ('position', ''),                ('goals', ''),
                   ('assists', ''),              ('minutes', ''),
              ('clean_sheets', ''), ('point_per_match', 'mean'),
                ('points', 'mean'),                 'ppm_value',
                         'p_value'],
      dtype='object')
```

[348]:
```
df_all = cost_df.merge(clean_t, "inner", on="name")

df_all["ppm_value"] = df_all[('point_per_match', 'mean')]/df_all.now_cost *1000
df_all["p_value"] = df_all[('points', 'mean')]/df_all.now_cost

df_all["ppm_value"] = df_all["ppm_value"].apply(lambda x:round(x,2))
df_all["p_value"] = df_all["p_value"].apply(lambda x:round(x,2))

df_all[ (df_all[('points', 'mean')] >= 90) & (df_all[('position', '')] !=␣
 ↪"DEF") \
      & (df_all[('position', '')] != "GK") ]\
.sort_values("ppm_value", ascending=False).head(59)
```

[348]:

	name	now_cost	(position,)	(goals,)	\
33	Riyad Mahrez	80	MID	11	
45	Emile Smith Rowe	60	MID	10	
27	Saïd Benrahma	60	MID	8	
83	Dejan Kulusevski	80	MID	5	
98	Manuel Lanzini	55	MID	5	

68	Emiliano Buendía Stati	60	MID	3
12	Mason Mount	80	MID	11
48	Ilkay Gündogan	75	MID	8
30	Harvey Barnes	70	MID	4
101	Armando Broja	55	FWD	6
8	James Maddison	80	MID	10
24	Conor Gallagher	60	MID	8
51	Gabriel Fernando de Jesus	80	FWD	8
94	Frederico Rodrigues de Paula Santos	55	MID	4
96	Alexis Mac Allister	55	MID	5
82	Anthony Gordon	55	MID	4
66	Jacob Ramsey	55	MID	6
72	Danny Ings	70	FWD	7
29	Phil Foden	80	MID	9
74	Demarai Gray	55	MID	5
70	Jorge Luiz Frello Filho	60	MID	6
19	Wilfried Zaha	70	MID	14
22	Leandro Trossard	65	MID	8
3	Jarrod Bowen	85	MID	10
63	Jordan Henderson	55	MID	2
61	Kai Havertz	80	MID	8
77	Che Adams	65	FWD	7
16	James Ward-Prowse	65	MID	10
13	Raheem Sterling	100	MID	13
4	Kevin De Bruyne	120	MID	15
9	Bukayo Saka	80	MID	11
86	Fabio Henrique Tavares	55	MID	5
34	Jamie Vardy	95	FWD	13
56	Jack Harrison	60	MID	8
0	Mohamed Salah	130	MID	23
38	Martin Ødegaard	65	MID	7
1	Heung-Min Son	120	MID	23
43	Rodrigo Hernandez	60	MID	7
21	Raphael Dias Belloli	70	MID	11
52	Youri Tielemans	65	MID	6
26	Ivan Toney	70	FWD	12
53	Bryan Mbeumo	60	MID	4
60	Tomas Soucek	55	MID	5
85	Neal Maupay	65	FWD	8
88	João Filipe Iria Santos Moutinho	50	MID	2
57	Allan Saint-Maximin	65	FWD	5
104	Jadon Sancho	75	MID	3
62	John McGinn	55	MID	3
25	Michail Antonio	75	FWD	10
64	Pierre-Emile Højbjerg	55	MID	2
15	Cristiano Ronaldo dos Santos Aveiro	105	FWD	18
102	Daniel James	60	MID	4

91	Declan Rice	50	MID	1	
79	Christian Nørgaard	55	MID	3	
36	Ollie Watkins	75	FWD	10	
103	Chris Wood	60	FWD	5	
80	Raúl Jiménez	70	FWD	6	
46	Richarlison de Andrade	85	FWD	8	
5	Harry Kane	115	FWD	16	

	(assists,)	(minutes,)	(clean_sheets,)	(point_per_match, mean)	\
33	6	1395	7	8.52	
45	2	1857	8	6.01	
27	6	2162	9	5.75	
83	9	1248	9	7.14	
98	3	1701	4	4.76	
68	6	1778	7	5.01	
12	11	2358	11	6.45	
48	5	1851	14	6.03	
30	11	1978	8	5.46	
101	2	1888	5	4.29	
8	9	2274	5	6.17	
24	6	2737	11	4.54	
51	8	1781	11	6.01	
94	5	2026	7	4.13	
96	2	2022	6	4.05	
82	3	2146	7	4.03	
66	1	2370	12	4.03	
72	6	1809	9	5.12	
29	6	2125	13	5.80	
74	4	2289	6	3.97	
70	2	2181	12	4.33	
19	1	2579	11	5.02	
22	4	2717	10	4.57	
3	17	2897	8	5.96	
63	5	2547	15	3.85	
61	4	1790	10	5.58	
77	4	2026	5	4.53	
16	5	3125	7	4.52	
13	7	2121	13	6.92	
4	8	2106	13	8.29	
9	9	2888	13	5.52	
86	1	2311	15	3.78	
34	3	1644	4	6.46	
56	1	2548	3	4.06	
0	14	2758	17	8.65	
38	4	2692	11	4.31	
1	10	2919	17	7.86	
43	1	2794	17	3.93	

21	4	2826	5	4.55
52	3	2539	6	4.11
26	5	2818	8	4.38
53	7	2815	9	3.74
60	1	2971	6	3.39
85	3	2269	7	3.89
88	1	2880	11	2.91
57	7	2728	8	3.76
104	4	1894	6	4.33
62	3	2910	10	3.09
25	9	2885	7	4.21
64	3	3106	15	3.07
15	3	2454	8	5.83
102	5	2489	5	3.29
91	4	3088	8	2.71
79	3	2964	8	2.95
36	3	2783	10	3.91
103	0	2694	6	3.04
80	5	2540	11	3.51
46	5	2342	7	4.07
5	11	3141	13	5.30

	(points, mean)	ppm_value	p_value
33	132.0	106.50	1.65
45	124.0	100.17	2.07
27	138.0	95.83	2.30
83	99.0	89.25	1.24
98	90.0	86.55	1.64
68	99.0	83.50	1.65
12	169.0	80.62	2.11
48	124.0	80.40	1.65
30	120.0	78.00	1.71
101	90.0	78.00	1.64
8	156.0	77.12	1.95
24	138.0	75.67	2.30
51	119.0	75.12	1.49
94	93.0	75.09	1.69
96	91.0	73.64	1.65
82	96.0	73.27	1.75
66	106.0	73.27	1.93
72	103.0	73.14	1.47
29	137.0	72.50	1.71
74	101.0	72.18	1.84
70	105.0	72.17	1.75
19	144.0	71.71	2.06
22	138.0	70.31	2.12
3	192.0	70.12	2.26

63	109.0	70.00	1.98
61	111.0	69.75	1.39
77	102.0	69.69	1.57
16	157.0	69.54	2.42
13	163.0	69.20	1.63
4	194.0	69.08	1.62
9	177.0	69.00	2.21
86	97.0	68.73	1.76
34	118.0	68.00	1.24
56	115.0	67.67	1.92
0	265.0	66.54	2.04
38	129.0	66.31	1.98
1	255.0	65.50	2.12
43	122.0	65.50	2.03
21	143.0	65.00	2.04
52	116.0	63.23	1.78
26	137.0	62.57	1.96
53	117.0	62.33	1.95
60	112.0	61.64	2.04
85	98.0	59.85	1.51
88	93.0	58.20	1.86
57	114.0	57.85	1.75
104	91.0	57.73	1.21
62	100.0	56.18	1.82
25	135.0	56.13	1.80
64	106.0	55.82	1.93
15	159.0	55.52	1.51
102	91.0	54.83	1.52
91	93.0	54.20	1.86
79	97.0	53.64	1.76
36	121.0	52.13	1.61
103	91.0	50.67	1.52
80	99.0	50.14	1.41
46	106.0	47.88	1.25
5	185.0	46.09	1.61

[282]:
```
len(df_all)
```

[282]:
```
181
```

[198]:
```
#all_df =␣
↪[cleaned_epl16_df,cleaned_epl17_df,cleaned_epl18_df,cleaned_epl19_df,cleaned_epl20_df,␣
↪cleaned_epl21_df]

all_df = [cleaned_epl20_df, cleaned_epl21_df]
```

```
main_df = pd.DataFrame()

for df in all_df:
    main_df = main_df.append(df)

lifetime_main_df = main_df.groupby(["name","position"], as_index=False)\
[['goals','assists', 'minutes','clean_sheets','ict', 'points']].sum()
```

[199]:
```
#inaccurate
lifetime_main_df["matches"] = lifetime_main_df.minutes/90
lifetime_main_df.matches = lifetime_main_df.matches.astype("int")

lifetime_main_df["goal_per_90"] = lifetime_main_df.goals/lifetime_main_df.
 ↪matches
lifetime_main_df.goal_per_90 = lifetime_main_df.goal_per_90.apply(lambda x:
 ↪round(x,2))

lifetime_main_df["assist_per_90"] = lifetime_main_df.assists/lifetime_main_df.
 ↪matches
lifetime_main_df.assist_per_90 = lifetime_main_df.assist_per_90.apply(lambda x:
 ↪round(x,2))

lifetime_main_df["points_per_90"] = lifetime_main_df.points/lifetime_main_df.
 ↪matches
lifetime_main_df.points_per_90 = lifetime_main_df.points_per_90.apply(lambda x:
 ↪round(x,2))

#lifetime_main_df["value"] = lifetime_main_df["points"]/lifetime_main_df["cost"]
#lifetime_main_df.value = lifetime_main_df.value.astype("int")

lifetime_main_df[(lifetime_main_df.points_per_90 >= 5.5) & (lifetime_main_df.
 ↪matches > 9)\
& (lifetime_main_df.position=="MID")].sort_values("points_per_90",␣
 ↪ascending=False).head()
```

[199]:

	name	position	goals	assists	minutes	clean_sheets	ict	points	matches	goal_per_90	assist_per_90	points_per_90
320	Gareth Bale	MID	11	3	912	5	136.4	113	10	1.10	0.30	11.30
684	Mohamed Salah	MID	45	20	5835	28	802.8	496	64	0.70	0.31	7.75
230	Dejan Kulusevski	MID	5	9	1248	9	106.9	99	13	0.38	0.69	7.62
800	Riyad Mahrez	MID	20	14	3340	18	410.4					
529	Kevin De Bruyne	MID	21	20	4101	25	568.9					

```
800     277     37      0.54            0.38            7.49
529     335     45      0.47            0.44            7.44
```

[ ]: