Percentile Analysis for ZCruit Prospects

Compilation of percentilies (increments of 10) in ZCruit's reported data for scouting events and other measurable variables for all positions in Zcruit database

In [3]:

```
import numpy as np
import pandas as pd
import zipfile
import os
```

Our first task is reading the csv from our ZCruit data export and creating a pandas dataframe with relevant details

In [4]:

```
athletes = pd.read_csv("Cleaned_Data.csv")
athletes = athletes[["Position", "Height", "Weight", "Forty Yard Dash", "Shuttle", "Vertic
athletes
```

```
/opt/conda/lib/python3.8/site-packages/IPython/core/interactiveshell.py:3165: DtypeWarnin
g: Columns (15,17,19,21,23,25,27,29,31,33) have mixed types.Specify dtype option on import
or set low_memory=False.
  has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

Out[4]:

	Position	Height	Weight	Forty Yard Dash	Shuttle	Vertical Jump	Broad Jump	3 Cone	Wingspan	Arm Length	Hand Size	Powerball Toss
0	DE	6' 4"	248.0	4.880	4.700	30.4	104.0	7.758	NaN	NaN	NaN	44.5
1	DE	6' 4"	275.0	4.750	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	OT	6' 6"	279.0	5.460	4.870	26.8	NaN	NaN	NaN	NaN	NaN	NaN
3	WR	6' 0.5"	190.0	4.859	4.200	35.1	112.0	7.083	75.0	NaN	9.00	41.0
4	DE	6' 2.5"	223.0	4.991	4.487	26.0	115.0	7.752	83.5	34.25	9.84	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
71365	WR	4' 8"	86.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
71366	WR	5' 1"	85.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
71367	WR	4' 11"	84.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
71368	WR	4' 10"	80.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
71369	RB	4' 9"	71.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

71370 rows × 12 columns

We define a function `create_percentile` that takes in two string arguements (position and event) that sorts the dataframe to only include athletes from the same position. We use the sorted dataframe to create a series containing the event results and convert those entries to functional values. After we create our series, we can create and returns a list to store the percentiles from the 0th to 100th incrementing by 10 for that position and event

In [5]:

```
def create_percentile(position, event):
    sorted_athletes = athletes[athletes['Position'] == position]
    event = (sorted_athletes[event].astype(float)).dropna()
    percentile = []
    for i in np.arange(0, 1.1, 0.1):
        percentile.append(event.quantile(i))
    return percentile
```

Below is an example of the function being used with Quaterback data in the 40-Yard Dash

In [6]:
```
create_percentile('QB', 'Forty Yard Dash')
```

Out[6]:
```
[4.1, 4.65, 4.7, 4.8, 4.85, 4.9, 5.0, 5.093, 5.2062000000000001, 5.3896, 54.9]
```

To simply the process of hard-coding every position for all the events, we create an array that has all the unique positions (stored as `unique_positions`) and an array of the events (stored as `events`)

In [7]:
```
unique_positions = athletes.Position.unique()
events = athletes.drop(['Position'], axis = 1).columns
```

In [8]:
```
unique_positions
```

Out[8]:
```
array(['DE', 'OT', 'WR', 'ATH', 'QB', 'TE', 'S', 'DT', 'LB', 'CB', 'RB',
       'OG', 'DB', 'DL', nan, 'OL', 'OC', 'K', 'LS', 'P', 'FB'],
      dtype=object)
```

In [9]:
```
events
```

Out[9]:
```
Index(['Height', 'Weight', 'Forty Yard Dash', 'Shuttle', 'Vertical Jump',
       'Broad Jump', '3 Cone', 'Wingspan', 'Arm Length', 'Hand Size',
       'Powerball Toss'],
      dtype='object')
```

Since our height is not in a functional format, we define a function `parse_ht` that is able to convert our entries as floats that we can use for our analysis

In [10]:
```
def parse_ht(ht):
    ht_ = ht.split("' ")
    ft_ = float(ht_[0])
    in_ = float(ht_[1].replace("\"",""))
    return (12*ft_) + in_
```

In [11]:
```
athletes['Height'] = athletes['Height'].apply(lambda x: parse_ht(x))
```

We want to create a dataframe of each event including all of the percentiles for each position. We do this by defining a function `create_df` with an arguement of the event and loops through every position to compute the percentile array and add it to a list that is converted to a dataframe

In [12]:
```
def create_df(event):
    percentile_list = []
    for i in unique_positions:
        percentile = create_percentile(i, event)
        percentile_list.append(percentile)
    return pd.DataFrame(data = percentile_list, index = unique_positions, columns = [i for
```

Below is an example of the function being used with the Shuttle event

In [14]:
```
create_df('Height')
```

Out[14]:

	0	10	20	30	40	50	60	70	80	90	100
DE	59.0	69.0	70.00	71.0	72.0	73.0	74.0	74.0	75.0	76.0	85.0
OT	63.0	71.0	72.25	74.0	75.0	75.0	76.0	77.0	77.0	78.0	83.0
WR	51.0	67.0	68.00	69.0	70.0	70.0	71.0	72.0	73.0	74.0	82.0
ATH	60.0	67.0	68.00	69.0	70.0	71.0	72.0	72.0	73.0	74.0	80.0
QB	57.0	68.0	70.00	70.0	71.0	72.0	72.0	73.0	74.0	75.0	80.0
TE	52.0	71.0	72.00	73.0	74.0	74.0	75.0	76.0	76.0	77.0	81.0
S	59.0	67.0	68.00	69.0	70.0	70.0	71.0	72.0	72.0	73.0	81.0
DT	57.0	68.0	69.00	70.0	71.0	72.0	73.0	74.0	74.0	75.0	80.0
LB	51.0	67.0	69.00	70.0	70.0	71.0	72.0	72.0	73.0	74.0	79.0
CB	57.0	66.0	67.00	68.0	69.0	69.0	70.0	71.0	71.0	72.0	79.0
RB	51.0	66.0	67.00	68.0	68.0	69.0	69.0	70.0	71.0	72.0	85.0
OG	60.0	68.0	70.00	71.0	72.0	73.0	74.0	74.0	75.0	76.0	83.0
DB	60.0	68.0	69.00	69.0	70.0	70.0	71.0	72.0	72.0	73.0	76.0
DL	62.0	70.0	71.00	72.0	72.0	73.0	74.0	74.0	75.0	75.0	80.0
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
OL	55.0	70.0	72.00	72.0	73.0	74.0	74.0	75.0	76.0	76.0	81.0
OC	61.0	68.0	69.00	70.0	71.0	72.0	73.0	73.0	74.0	75.0	79.0
K	61.0	68.0	69.00	69.0	70.0	71.0	71.0	72.0	73.0	74.0	80.0
LS	66.0	69.0	70.00	70.0	71.0	72.0	72.0	73.0	74.0	75.0	77.0
P	65.0	69.0	70.00	71.0	72.0	72.0	73.0	74.0	74.0	75.0	78.0
FB	60.0	65.0	67.00	68.0	68.0	69.0	70.0	70.0	72.0	73.0	75.0

We create a function `to_csv` and use the dataframe that we made and covert it into a csv and save it to our Jupyter Folder

In [13]:

```
def to_csv(event):
    df_event = create_df(event)
    df_event.to_csv(r''+event+'Percentile.csv')
```

We can run our function for all our events and obtain the percentile csv for all of the events and positions

In [ ]:

```
for i in events:
    to_csv(i)
```