```
In [2]: # importing my library
        import matplotlib.pyplot as plt
        import seaborn as sns
        import numpy as np
        import pandas as pd
        from sklearn.preprocessing import MinMaxScaler
```

```
In [ ]:
```

```
In [3]: # imported my CSV file.
        df = pd.read_csv(r'C:\Users\hp\Documents\Muskets_teamData_V2.csv')
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_10820\2088104513.py:1: DtypeWarning: Columns (26,29,76) have mixed types. Spec
ify dtype option on import or set low_memory=False.
  df = pd.read_csv(r'C:\Users\hp\Documents\Muskets_teamData_V2.csv')
```

```
In [4]: df
```

Out[4]:

	ID	Name	LongName	photoUrl	playerUrl	Nationality	Age
0	158023	L. Messi	Lionel Messi	https://cdn.sofifa.com/players/158/023/21_60.png	http://sofifa.com/player/158023/lionel-messi/2...	Argentina	33
1	20801	Cristiano Ronaldo	C. Ronaldo dos Santos Aveiro	https://cdn.sofifa.com/players/020/801/21_60.png	http://sofifa.com/player/20801/c-ronaldo-dos-s...	Portugal	35
2	200389	J. Oblak	Jan Oblak	https://cdn.sofifa.com/players/200/389/21_60.png	http://sofifa.com/player/200389/jan-oblak/210006/	Slovenia	27
3	192985	K. De Bruyne	Kevin De Bruyne	https://cdn.sofifa.com/players/192/985/21_60.png	http://sofifa.com/player/192985/kevin-de-bruyn...	Belgium	29
4	190871	Neymar Jr	Neymar da Silva Santos Jr.	https://cdn.sofifa.com/players/190/871/21_60.png	http://sofifa.com/player/190871/neymar-da-silv...	Brazil	28
...	...	...	...	...	...	...	...
19016	247223	Xia Ao	Ao Xia	https://cdn.sofifa.com/players/247/223/21_60.png	http://sofifa.com/player/247223/ao-xia/210006/	China PR	21
19017	258760	B. Hough	Ben Hough	https://cdn.sofifa.com/players/258/760/21_60.png	http://sofifa.com/player/258760/ben-hough/210006/	England	17
19018	252757	R. McKinley	Ronan McKinley	https://cdn.sofifa.com/players/252/757/21_60.png	http://sofifa.com/player/252757/ronan-mckinley...	England	18
19019	243790	Wang Zhen'ao	Zhen'ao Wang	https://cdn.sofifa.com/players/243/790/21_60.png	http://sofifa.com/player/243790/zhenao-wang/21...	China PR	20
19020	252520	Zhou Xiao	Xiao Zhou	https://cdn.sofifa.com/players/252/520/21_60.png	http://sofifa.com/player/252520/xiao-zhou/210006/	China PR	21

19021 rows × 77 columns

In [ ]:

In [5]:

```
# TASK 1
    #Extract the player names from the playerurl column and create a new column name Player
    #  Name from the extracts
```

```
In [6]: df.playerUrl.values
```

```
Out[6]: array(['http://sofifa.com/player/158023/lionel-messi/210006/',
               'http://sofifa.com/player/20801/c-ronaldo-dos-santos-aveiro/210006/',
               'http://sofifa.com/player/200389/jan-oblak/210006/', ...,
               'http://sofifa.com/player/252757/ronan-mckinley/210006/',
               'http://sofifa.com/player/243790/zhenao-wang/210006/',
               'http://sofifa.com/player/252520/xiao-zhou/210006/'], dtype=object)
```

```
In [7]: playerList = []

        for item in df.playerUrl.values:
            player = item.split('/')[-3]
            player = player.replace('-','')
            playerList.append(player)
```

```
In [8]: playerList
```

```
Out[8]: ['lionelmessi',
 'cronaldodossantosaveiro',
 'janoblak',
 'kevindebruyne',
 'neymardasilvasantosjr',
 'robertlewandowski',
 'mohamedsalah',
 'alissonramsesbecker',
 'kylianmbappe',
 'marcandreterstegen',
 'virgilvandijk',
 'sadiomane',
 'carloshenriquevenanciocasimiro',
 'thibautcourtois',
 'manuelneuer',
 'karimbenzema',
 'sergioramosgarcia',
 'sergioaguero',
 'raheemsterling',
 'ngolokante',
 'joshuakimmich',
 'paulodybala',
 'edersonsantanademoraes',
 'harrykane',
 'samirhandanovic',
 'kalidoukoulibaly',
 'edenhazard',
 'tonikroos',
 'antoinegriezmann',
 'jadonsancho',
 'trentalexanderarnold',
 'bernardomotacarvalhoesilva',
 'andrewrobertson',
 'aymericlaporte',
 'brunomiguelborgesfernandes',
 'fabiohenriquetavares',
 'heungminson',
 'robertofirminobarbosadeoliveira',
 'keylornavas',
 'giorgiochiellini',
 'sergiobusquetsburgos',
 'pierreemerickaubameyang',
 'wojciechszczesny',
 'angeldimaria',
 'lukamodric',
 'luissuarez',
 'hugolloris',
 'ciroimmobile',
 'thomasmuller',
 'daviddegeaquintana',
 'danielcarvajalramos',
 'raphaelvarane',
 'marcoverratti',
 'paulpogba',
 'jamievardy',
 'gerardpiquebernabeu',
 'matshummels',
 'yannsommer',
 'davidjosuejimenezsilva',
 'alejandrogomez',
 'jordialbaramos',
 'jordanhenderson',
 'rodrigohernandezcascante',
 'sergejmilinkovicsavic',
 'memphisdepay',
 'kaihavertz',
 'matthijsdeligt',
 'milanskriniar',
 'marcusrashford',
 'gianluigidonnarumma',
 'sergegnabry',
 'leroysane',
 'hakimziyech',
 'clementlenglet',
 'marcosaoascorrea',
 'riyadmahrez',
 'ricardobarbosapereira',
 'timowerner',
 'danielparejomunoz',
 'mauroicardi',
 'frenkiedejong',
 'lorenzoinsigne',
 'tobyalderweireld',
 'thiagoemilianodasilva',
 'jorgeresurreccion',
 'berndleno',
 'romelulukaku',
 'alexsandrolobosilva',
 'thiagoalcantara',
 'kylewalker',
 'christianeriksen',
 'petergulacsi',
 'leonardobonucci',
 'driesmertens',
 'luisalbertoromeroalconchel',
 'diegogodin',
 'georginiowijnaldum',
 'miralempjanic',
 'marcoreus',
 'wilfredndidi',
 'josemariagimenez',
 'andreonana',
 'julianbrandt',
 'mikeloyarzabalugarte',
 'lautaromartinez',
 'erlinghaaland',
 'alexnicolaotelles',
 'kingsleycoman',
 'alejandrogrimaldogarcia',
 'arthurhenriqueramosoliveiramelo',
 'niklassule',
 'raphaelguerreiro',
 'anthonymartial',
 'felipeaugustodealmeidamonteiro',
 'leongoretzka',
 'saulniguezesclapez',
 'thomaspartey',
 'wissambenyedder',
 'nicolastagliafico',
 'rauljimenez',
 'marcelobrozovic',
 'lucasdigne',
 'romanburki',
 'josipilicic',
 'josemariacallejonbueno',
 'luismiguelafonsofernandes',
 'franciscoromanalarconsuarez',
 'davidalaba',
 'idrissagueye',
 'jesuscorona',
 'iagoaspasjuncal',
 'douglascostadesouza',
 'stefandevrij',
 'fernandoluizrosa',
 'lucaspezzinileiva',
 'cesarazpilicuetatanco',
 'salvatoresirigu',
 'kasperschmeichel',
 'axelwitsel',
 'ruipedrodossantospatricio',
 'edinsoncavani',
 'jesusnavasgonzalez',
 'achrafhakimi',
 'ferlandmendy',
 'ousmanedembele',
 'thorganhazard',
 'ronaldojailsoncabraispetri',
 'aaronwanbissaka',
 'deniszakaria',
 'nabilfekir',
 'marcosacuna',
 'stevenbergwijn',
 'joegomez',
 'martinodegaard',
 'donnyvandebeek',
 'paulopezsabata',
 'martindubravka',
 'diegocarlossantossilva',
 'federicovalverde',
 'gerardmorenobalaguero',
 'gabrielfernandodejesus',
 'rafaelaferreirasilva',
 'lucasrodriguesmsilva',
 'thomasstrakosha',
 'filipkostic',
 'joseluisgayapena',
 'delealli',
 'joaopedrocavacocancelo',
 'alessioromagnoli',
 'juanbernatvelasco',
 'edinvisca',
 'samuelumtiti',
 'luizfrellofilhojorge',
 'cristianportuguesmanzanera',
 'marcelsabitzer',
 'duvanzapata',
 'mateokovacic',
 'nelsoncabralsemedo',
 'fernandoreges',
 'dusantadic',
 'blaisematuidi',
 'kierantrippier',
 'sergiorobertocarnicer',
 'anthonylopes',
 'charlesaranguiz',
 'joelmatip',
 'aitorfernandez',
 'alexandrelacazette',
 'wilfriedzaha',
 'kostasmanolas',
 'lukashradecky',
 'ikermuniaingoni',
 'philippecoutinhocorreia',
 'oscardossantosemboaba',
 'koencasteels',
 'francescoacerbi',
 'ilkaygundogan',
 'kevintrapp',
 'josepaulobezerramjunior',
 'yuriberchicheizeta',
 'allanmarquesloureiro',
 'arturovidal',
 'edindzeko',
 'radjanainggolan',
 'marcelovieiradasilva',
 'garethbale',
 'everbanega',
 'janvertonghen',
 'stevemandanda',
 'zlatanibrahimovic',
 'carlosvela',
 'joaofilipemoutinho',
 'gonzalohiguain',
 'renatoaugusto',
 'lucashernandez',
 'dakonamdjene',
 'florianthauvin',
 'jiripavlenka',
 'josuedurvalchiamuleravaz',
 'giovanilocelso',
 'marcosllorentemoreno',
 'nicolaspepe',
 'konradlaimer',
 'rubendiogodasilvaneves',
 'davidsoriasolis',
 'robingosens',
 'marcoasensiowillemsen',
 'yannickcarrasco',
 'sergioreguilonrodriguez',
 'andrejkramaric',
 'quincypromes',
 'mikemaignan',
 'lucasocampos',
 'kepaarrizabalaga',
 'emrecan',
 'matthiasginter',
 'fernandopachecoflores',
 'marcelhalstenberg',
 'andersonsouzaconceicao',
 'angelcorrea',
 'geronimorulli',
 'fabianruizpena',
 'tomasvaclik',
 'raphaelwilliamanjosrochedo',
 'harrymaguire',
 'daleyblind',
 'nickpope',
 'rodrigomorenomachado',
 'alphonseareola',
 'mariofigueirafernandes',
 'norbertomuraraneto',
 'mattiaperin',
 'agustinmarchesin',
 'jamesrodriguez',
 'daniloluisheliopereira',
 'alvaroborjamoratamartin',
 'gabrielarmandodeabreu',
 'sergiocanalesmadrazo',
 'karimbellarabi',
 'oliverbaumann',
 'sergioasenjoandres',
 'larsbender',
 'gianluigibuffon',
 'aaronramsey',
 'raulalbioltortajada',
 'mesutozil',
 'dimitripayet',
 'ivanrakitic',
 'willianborgesdasilva',
 'markoarnautovic',
 'fernandomuslera',
 'aleksandarkolarov',
 'jeromeboateng',
 'ivanperisic',
 'houssemaouar',
 'manuellazzari',
 'marccucurellasaseta',
 'rubensantosgatoalvesdias',
 'moussadiaby',
 'ferrantorresgarcia',
 'joaofelixsequeira',
 'renanaugustolodidossantos',
 'raphaeldiasbelloli',
 'josefmartinez',
 'alphonsodavies',
 'rosbertojdouradosantos',
 'richarlisondeandrade',
 'marvinoswaldorangelazevedo',
 'nunoailtonpadrendamendes',
 'welingtonkauedanonascimento',
 'lourivaladnanberettabarbosa',
 'adryanjulianozontatorres',
 'juanevertonmestresdemesquita',
 'benchilwell',
 'christianpulisic',
 'tomassoucek',
 'markodmitrovic',
 'woutweghorst',
 'presnelkimpembe',
 'andercaparodriguez',
 'lukeshaw',
 'youritielemans',
 'matteopolitano',
 'jamesmaddison',
 'nabykeita',
 'evertonsousasoares',
 'lucastorreira',
 'inakiwilliamsarthuer',
 'davinsonsanchez',
 'jonathancastrootto',
 'ignaciofernandez',
 'joaquincorrea',
 'walterbenitez',
 'thomaslemar',
 'odisseasvlachodimos',
 'moussadembele',
 'adrienrabiot',
 'fredericodepaulasantos',
 'goncalomanuelganchinhoguedes',
 'martenderoon',
 'benjaminpavard',
 'thomasdelaney',
 'jordimasiplopez',
 'xherdanshaqiri',
 'sebastiancoates',
 'kevinkampl',
 'alessiocragno',
 'antoniorudiger',
 'arkadiuszmilik',
 'jordanpickford',
 'stefansavic',
 'eranzahavi',
 'santiagoarias',
 'alessandroflorenzi',
 'jesusjoaquinfernandezsaez',
 'benjaminmendy',
 'felipeandersonpereiragomes',
 'mattdoherty',
 'kevinvolland',
 'franciscoalcacergarcia',
 'stevenberghuis',
 'jaspercillessen',
 'pablosarabiagarcia',
 'thomasmeunier',
 'sebastienhaller',
 'simonmignolet',
 'ismailygoncalvesdoss',
 'jonathanvieraramos',
 'svenbender',
 'juancuadrado',
 'andresiniestalujan',
 'marcosalonsomendoza',
 'mariogotze',
 'eduardosalvio',
 'marlosromerobonfim',
 'maxkruse',
 'samikhedira',
 'javiermartinezaginaga',
 'taisonbarcellosfreda',
 'ivanmarcanosierra',
 'marekhamsik',
 'stephaneruffier',
 'lukaszfabianski',
 'andreaconsigli',
 'franckribery',
 'hectorherrera',
 'keplerlaveranlimaferreira',
 'estebanandrada',
 'andrepierregignac',
 'jaimenicolasfrendado',
 'viniciusjosedeoliveirajunior',
 'maximilianogomez',
 'dominiklivakovic',
 'viktortsygankov',
 'jorgeezequielserendero',
 'mauroevidionerez',
 'matiasdavidbaldona',
 'saulmarceloardero',
 'luisrobertodalves',
 'egidiomaestreschetino',
 'josemariasildero',
 'tanguyndombele',
 'bendavies',
 'davidnerescampos',
 'christophernkunku',
 'deanhenderson',
 'caglarsoyuncu',
 'edergabrielmilitao',
 'unaisimonmendibil',
 'masonmount',
 'mauroarambarri',
 'theohernandez',
 'lukajovic',
 'tonyjosimarabranjesmeneses',
 'marcusthuram',
 'oleksandrzinchenko',
 'laureanothomassanteiromarre',
 'segundomandiquez',
 'maikelrenancatarinofagundes',
 'leonbailey',
 'mariohermosocanseco',
 'yerayalvarezlopez',
 'nordimukiele',
 'alejandroremirogargallo',
 'stefanlainer',
 'danielceballosfernandez',
 'mikelmerinozazon',
 'allansaintmaximin',
 'henitonenaldopirestramontino',
 'diogojoseteixeiradasilva',
 'nicolobarella',
 'valentinrongier',
 'otavioedmilsondasilvamonteiro',
 'lukasklostermann',
 'hirvinglozano',
 'victorlindelof',
 'joseangelesmoristasende',
 'matiasvecino',
 'corentintolisso',
 'juanmusso',
 'piotrzielinski',
 'pierreemilehojbjerg',
 'alassaneplea',
 'gelsondanybatalhamartins',
 'jackgrealish',
 'williamsilvadecarvalho',
 'adnanjanuzaj',
 'andreabelotti',
 'francoarmani',
 'lucasvazqueziglesias',
 'domenicoberardi',
 'rubenpenajimenez',
 'danielerugani',
 'keremdemirbay',
 'federicobernardeschi',
 'joaomarionavalcostaeduardo',
 'inigomartinezberridi',
 'nikolavlasic',
 'simevrsaljko',
 'juliandraxler',
 'granitxhaka',
 'josegomezcampana',
 'nicoschulz',
 'joseignaciofernandeziglesias',
 'benjaminlecomte',
 'martinhinteregger',
 'gerarddeulofeulazaro',
 'ademljajic',
 'jeromeroussillon',
 'willyboly',
 'johnstones',
 'pavelkaderabek',
 'hectorbellerinmoruno',
 'damiansuarez',
 'jamestarkowski',
 'cedricbakambu',
 'asierillarramendi',
 'mathewryan',
 'benjaminandre',
 'ricardogoulartpereira',
 'mariogasparperezmartinez',
 'henrikhmkhitaryan',
 'sergioescuderopalomo',
 'willianjosedasilva',
 'dannyings',
 'marcbartraaregall',
 'enzoperez',
 'grzegorzkrychowiak',
 'andregomesmagalhaesalmeida',
 'sergeaurier',
 'givanildovieiradesouza',
 'luukdejong',
 'nemanjamatic',
 'joaomirandadesouzafilho',
 'sebastiangiovinco',
 'ignaciomonrealeraso',
 'antonioadangarrido',
 'branislavivanovic',
 'simonkjaer',
 'diegodasilvacosta',
 'davidluizmoreiramarinho',
 'giulianovictordepaula',
 'sergioromero',
 'luizgustavodias',
 'larsstindl',
 'nicolasnkoulou',
 'giacomobonaventura',
 'vicenteguaitapanadero',
 'franciscoquelin',
 'lukaszpiszczek',
 'alexissanchez',
 'kurtzouma',
 'bafetimbigomis',
 'danielwass',
 'arjenrobben',
 'raulgarciaescudero',
 'eriklamela',
 'radamelfalcaogarciazarate',
 'mousadembele',
 'jonnyevans',
 'pedrorodriguezledesma',
 'igorakinfeev',
 'luiscarlosalmeidadacunha',
 'jamesmilner',
 'joaquinsanchezrodriguez',
 'fabioquagliarella',
 'leandromiguelsareda',
 'aluisiochavesribmoraesjunior',
 'enzomartinriquero',
 'rafaeldileonardo',
 'edgarmanuelguichon',
 'morgansanson',
 'carlosviniciusalvesmorais',
 'danielolmocarvajal',
 'rodrygosilvadegoes',
 'danaxelzagadou',
 'predragrajkovic',
 'juleskounde',
 'hectorjuniorfirpoadames',
 'youcefatal',
 'nemanjamaksimovic',
 'pervisestupinan',
 'philfoden',
 'enisbardhi',
 'sanderberge',
 'scottmctominay',
 'florianneuhaus',
 'denzeldumfries',
 'harveybarnes',
 'jonathanikone',
 'boubacarkamara',
 'declanrice',
 'carlossolerbarragan',
 'alvaroodriozolaarzalluz',
 'igorzubeldiaelorza',
 'alexanderisak',
 'victorosimhen',
 'ruslanmalinovskyi',
 'luanvagnerboasmacedo',
 'bernardgabrielprestaochaves',
 'melvinluanparrelaporfirio',
 'evertonjorgeandradezanon',
 'adnanclaudianovidualmachado',
 'ronaldodiegoeslerdomingues',
 'dayotupamecano',
 'milotrashica',
 'andremiguelvalentedasilva',
 'kasperdolberg',
 'ezequielavila',
 'lorenzopellegrini',
 'jaimemataarnaiz',
 'moussamarega',
 'leodubois',
 'pablofornalsmalla',
 'alexandrgolovin',
 'hamaritraore',
 'ericbailly',
 'dujecaletacar',
 'joanjordanmoreno',
 'cristianpavon',
 'borjaiglesiasquintas',
 'petrosmatheusdossantosaraujo',
 'ramybensebaini',
 'tiemouebakayoko',
 'dominiccalvertlewin',
 'munirelhaddadi',
 'jasondenayer',
 'josebazalduabengoetxea',
 'anterebic',
 'julianweigl',
 'baptistesantamaria',
 'emersonpalmieridossantos',
 'harrywinks',
 'adamatraorediarra',
 'joaopaulodiasfernandes',
 'joseluismoralesnogales',
 'danielgarciacarrillo',
 'armandoizzo',
 'ayozeperezgutierrez',
 'miltoncasco',
 'dariobenedetto',
 'wilmarbarrios',
 'alexeymiranchuk',
 'frankfabra',
 'abderrazakhamdallah',
 'andreaschristensen',
 'ricardojorgedaluzhorta',
 'jonathantah',
 'josepedromalheirodesa',
 'philippmax',
 'andrefilipetavaresgomes',
 'pierluigigollini',
 'valentinolazaro',
 'rodrigobentancur',
 'rubengarciasantos',
 'edouardmendy',
 'luismuriel',
 'alejandropozuelomelero',
 'fernandolucasmartins',
 'nicolasotamendi',
 'ruitiagodantasdasilva',
 'nathanake',
 'omarmascarellgonzalez',
 'munasdabbur',
 'emilforsberg',
 'hakancalhanoglu',
 'matheuslimamagalhaes',
 'ryanfraser',
 'leandroparedes',
 'maximilianarnold',
 'tarasstepanenko',
 'lukamilivojevic',
 'jameswardprowse',
 'williorban',
 'marioruisilvaduarte',
 'kennylala',
 'andrecarrillo',
 'alvarogonzalezsoberon',
 'alexanderschwolow',
 'gabrielappeltpires',
 'pedroleonsanchezgil',
 'kesenai',
 'samuelcastillejoazuaga',
 'ibaigomezperez',
 'faouzighoulam',
 'rafaelalcantara',
 'jordanveretout',
 'aissamandi',
 'marcobizot',
 'victormachinperez',
 'matzsels',
 'nicolaslodeiro',
 'lucasperezmartinez',
 'daniloluizdasilva',
 'joaopedrogsantosgalvao',
 'geoffreykondogbia',
 'alexoxladechamberlain',
 'andydelort',
 'nemanjagudelj',
 'jaumevicentcostajorda',
 'conorcoady',
 'fabianorellana',
 'salifsane',
 'anderherreraaguera',
 'stephanelshaarawy',
 'joseangelvaldesdiaz',
 'chrissmalling',
 'benmee',
 'vicenteiborradelafuente',
 'francescocaputo',
 'alexteixeiradossantos',
 'stevennzonzi',
 'artemdzyuba',
 'svenulreich',
 'cristhianstuani',
 'sofianefeghouli',
 'gylfisigurdhsson',
 'diegoperotti',
 'marcoparolo',
 'moussasissoko',
 'nordinamrabat',
 'sidneirecheldasilvajunior',
 'vitorinopachecoantunes',
 'vedrancorluka',
 'seamuscoleman',
 'yannmvila',
 'oliviergiroud',
 'dannyrose',
 'angelluisrodriguezdiaz',
 'juanmanuelmatagarcia',
 'mathieuvalbuena',
 'marcelodiaz',
 'nahuelguzman',
 'kevingameiro',
 'kevinmbabu',
 'sokratispapastathopoulos',
 'diegovaleri',
 'davidospina',
 'laurentkoscielny',
 'benfoster',
 'benoitcostil',
 'javiermascherano',
 'guillermoochoa',
 'josemanuelreinapaez',
 'ricardoandradequaresmabernardo',
 'marciorafaelferreiradesouza',
 'edercitadinmartins',
 'antonymatheusdossantos',
 'enzomanuelaguerro',
 'vladimircoufal',
 'franciscomotacastrotrincao',
 'danielmartinlenzado',
 'brunoguimaraesmoura',
 'sebastianbrunoluna',
 'emersonleitedesouza',
 'edmondtapsoba',
 'luismanuelarantesmaximiano',
 'ondrejkolar',
 'lisandromartinez',
 'dwightmcneil',
 'lorenzomorongarcia',
 'anthonynwakaeme',
 'paufranciscotorres',
 'luisdiaz',
 'unainunezgestoso',
 'teunkoopmeiners',
 'matheussantoscarneirodacunha',
 'oscarmelendojimenez',
 'ibrahimakonate',
 'matiasrojas',
 'stefanosensi',
 'calvinstengs',
 'albanlafont',
 'federicochiesa',
 'matheusjadsonbardeiracunha',
 'cengizunder',
 'domingossousamenezesduarte',
 'patrikschick',
 'gabrieldossmagalhaes',
 'donyellmalen',
 'tammyabraham',
 'gonzalomontiel',
 'miguelalmiron',
 'renatojuniorluzsanches',
 'jonathanprazeresconradi',
 'fabianschar',
 'ismailasarr',
 'fredclaudineitofolivilela',
 'arturoadolfoinalciodutra',
 'gpierrepaivasouza',
 'albertomarcelodutracirino',
 'evertonjorgesimaoresende',
 'vincentefilhodouradohermes',
 'abenjaminchiamuloirapaes',
 'wellingtedsonsabraorolim',
 'gersonadrianogutierresserra',
 'falayesacko',
 'pierreleesmelou',
 'breelembolo',
 'joaomariapalhinhagoncalves',
 'jordanamavi',
 'floriangrillitsch',
 'danielcastelopodence',
 'kierantierney',
 'benjaminverbic',
 'abdoudiallo',
 'alexmeret',
 'jonathanbamba',
 'paulbernardoni',
 'jonathanrodriguez',
 'rubenaguilar',
 'krzysztofpiatek',
 'karltokoekambi',
 'jaumedomenechsanchez',
 'hanshateboer',
 'jonathanrodriguezmenendez',
 'jeanpierrensame',
 'manuellocatelli',
 'nicoelvedi',
 'leiwu',
 'rafaelsantosborre',
 'yerrymina',
 'mattiacaldara',
 'angelmena',
 'francocervi',
 'marcospaulomesquitalopes',
 'samuelgigot',
 'nunomigueljeronimosequeira',
 'stanislavlobotka',
 'marceloaugustoferreirateixeira',
 'davidezappacosta',
 'ivanmarcone',
 'jeanphilippegbamin',
 'carlosizquierdoz',
 'benjaminbourigeaud',
 'mateusuribe',
 'rubenloftuscheek',
 'romanzobnin',
 'leanderdendoncker',
 'jackoconnell',
 'ricardosousaesgaio',
 'joseignaciomartinezgarcia',
 'robertogagliardini',
 'vincenzogrifo',
 'floriannniederlechner',
 'elseidhysaj',
 'renatosteffen',
 'darwinmachis',
 'rauldetomasgomez',
 'jhoncordoba',
 'manuelakanji',
 'youssefennesyri',
 'suatserdar',
 'miguelangelmoyarumbo',
 'seadkolasinac',
 'christophkramer',
 'burakyilmaz',
 'hirokisakai',
 'romainsaiss',
 'bryancristante',
 'abdoulayedoucoure',
 'yussufpoulsen',
 'michaelkeane',
 'robinolsen',
 'lucianovietto',
 'leandrotrossard',
 'manueltriguerosmunoz',
 'mattiadesciglio',
 'leonardocarrilhobaptistao',
 'zouhairfeddal',
 'tejisavanier',
 'bernardaniciocaldeiraduarte',
 'filipnovak',
 'denissuarezfernandez',
 'takashiinui',
 'johnegan',
 'michybatshuayi',
 'janboril',
 'loriskarius',
 'emilianomartinez',
 'younesbelhanda',
 'maiconpereiraroque',
 'florentmollet',
 'djibrilsidibe',
 'hansvanaken',
 'cristiantelloherrera',
 'nacerchadli',
 'dannydacosta',
 'ericdier',
 'pablonascimentocastro',
 'cipriantatarusanu',
 'remofreuler',
 'diegodemme',
 'nicolaspallois',
 'lewisdunk',
 'patrickherrmann',
 'domagojvida',
 'rossbarkley',
 'kevinmalcuit',
 'nathanredmond',
 'fredrikmidtsjo',
 'hugomallonovegil',
 'tomheaton',
 'oscardemarcosarana',
 'leonardospinazzola',
 'mbayeniang',
 'jonassvensson',
 'martinskrtel',
 'basdost',
 'christiangunter',
 'ryanbertrand',
 'guilhermealvimmarinato',
 'gervaisyaokouassi',
 'dieumercimbokani',
 'celsoortiz',
 'stefanradu',
 'ignaciopiatti',
 'cristianansaldi',
 'rogeliofunesmori',
 'michailantonio',
 'sergiomiguelrelvasdeoliveira',
 'marwinhitz',
 'victorlaguardiacisneros',
 'youssefelarabi',
 'germanpezzella',
 'ricardorodriguez',
 'iagoherrerinbuisan',
 'endastevens',
 'chriswood',
 'alexandrkokorin',
 'shkodranmustafi',
 'luisrodriguez',
 'ashleywestwood',
 'callumwilson',
 'manuellanzini',
 'kevinprinceboateng',
 'senadlulic',
 'antoniomirante',
 'lautaroacosta',
 'matiassuarez',
 'rafaeldesouzapereira',
 'mateomusacchio',
 'josuhaguilavogui',
 'victordaviddiazmiguel',
 'papakoulidiop',
 'etiennecapoue',
 'felipecaicedo',
 'pablopiatti',
 'antoniocandreva',
 'mathieudebuchy',
 'carlostevez',
 'domenicocriscito',
 'josesosa',
 'jorgemolinavidal',
 'andreseduardofernandezmoreno',
 'josemigueldarochafonte',
 'andresguardado',
 'victormoses',
 'lisandrolopez',
 'gastonramirez',
 'lucaslucianomantelapatricio',
 'timothycastagne',
 'robertopereyra',
 'layvinkurzawa',
 'isaachayden',
 'harrywilson',
 'owenwijndal',
 'mariopasalic',
 'luigisepe',
 'robertsnodgrass',
 'franckyannickkessie',
 'masonholgate',
 'hassanekamara',
 'federicofernandez',
 'nadiemamiri',
 'wendersonnascimentogaleno',
 'waynerooney',
 'heechanhwang',
 'mehditaremi',
 'manuelhtavaresfernandes',
 'thomasfoket',
 'claudionunocointracalegari',
 'luismontes',
 'kevinvogt',
 'hendrikvancrombrugge',
 'danilodambrosio',
 'mariolemina',
 'rafalgikiewicz',
 'adammarusic',
 'sidneyadrianpessoavidigal',
 'olivernorwood',
 'aridanehernandezumpierrez',
 'ivancuellarsacristan',
 'sebastianrudy',
 'arthurmasuaku',
 'yuriyzhirkov',
 'sergienrichametller',
 'salomonrondon',
 'ricardocenturion',
 'franciscoreisferreira',
 'takumiminamino',
 'juliobuffarini',
 'gautierlarsonneur',
 'johnathanaparecidosilva',
 'ralffahrmann',
 'nealmaupay',
 'antoniojoserodriguezdiaz',
 'luizfeliperamosmarchi',
 'daleysinkgraven',
 'jordanayew',
 'jonasomlin',
 'henryonyekuru',
 'emilaudero',
 'stevanjovetic',
 'angeloogbonna',
 'vladimirdarida',
 'mikelsanjosedominguez',
 'gabrielarias',
 'marcosilvestri',
 'ashleybarnes',
 'simoneverdi',
 'fransergiorodriguesbarbosa',
 'sebastianrode',
 'divockorigi',
 'ozankabak',
 'kevinstrootman',
 'mattritchie',
 'timohorn',
 'rubenmiguelnunesvezo',
 'pedrohenriquepereiradasilva',
 'patrickvanaanholt',
 'joshuaking',
 'enzoleandrochissanobaia',
 'rubenduartesanchez',
 'noussairmazraoui',
 'damiendasilva',
 'vicentedamiancastro',
 'bartolomeuquissanga',
 'nikolakalinic',
 'davidsondaluzpereira',
 'kalvinphillips',
 'robertosoriano',
 'adamlallana',
 'martinmontoyatorralbo',
 'germansanchezbarahona',
 'leonardobittencourt',
 'regisgurtner',
 'claudiopiresdemoraisramos',
 'francovazquez',
 'shaneduffy',
 'odionighalo',
 'brunodasilvaperes',
 'matheusleitenascimento',
 'ellyesskhiri',
 'islamslimani',
 'dejankulusevski',
 'andreassamaris',
 'camilovargas',
 'robertorosales',
 'johannberggudhmundsson',
 ...]
```

In [9]:
```
# Adding playername to my dataframe.
df['playerName'] = playerList
```

In [10]:
```
df
```

Out[10]:

	ID	Name	LongName	photoUrl	playerUrl	Nationality	Age
0	158023	L. Messi	Lionel Messi	https://cdn.sofifa.com/players/158/023/21_60.png	http://sofifa.com/player/158023/lionel-messi/2...	Argentina	33
1	20801	Cristiano Ronaldo	C. Ronaldo dos Santos Aveiro	https://cdn.sofifa.com/players/020/801/21_60.png	http://sofifa.com/player/20801/c-ronaldo-dos-s...	Portugal	35
2	200389	J. Oblak	Jan Oblak	https://cdn.sofifa.com/players/200/389/21_60.png	http://sofifa.com/player/200389/jan-oblak/210006/	Slovenia	27
3	192985	K. De Bruyne	Kevin De Bruyne	https://cdn.sofifa.com/players/192/985/21_60.png	http://sofifa.com/player/192985/kevin-de-bruyn...	Belgium	29
4	190871	Neymar Jr	Neymar da Silva Santos Jr.	https://cdn.sofifa.com/players/190/871/21_60.png	http://sofifa.com/player/190871/neymar-da-silv...	Brazil	28
...	...	...	...	...	...	...	...
19016	247223	Xia Ao	Ao Xia	https://cdn.sofifa.com/players/247/223/21_60.png	http://sofifa.com/player/247223/ao-xia/210006/	China PR	21
19017	258760	B. Hough	Ben Hough	https://cdn.sofifa.com/players/258/760/21_60.png	http://sofifa.com/player/258760/ben-hough/210006/	England	17
19018	252757	R. McKinley	Ronan McKinley	https://cdn.sofifa.com/players/252/757/21_60.png	http://sofifa.com/player/252757/ronan-mckinley...	England	18
19019	243790	Wang Zhen'ao	Zhen'ao Wang	https://cdn.sofifa.com/players/243/790/21_60.png	http://sofifa.com/player/243790/zhenao-wang/21...	China PR	20
19020	252520	Zhou Xiao	Xiao Zhou	https://cdn.sofifa.com/players/252/520/21_60.png	http://sofifa.com/player/252520/xiao-zhou/210006/	China PR	21

19021 rows × 78 columns

In [ ]:

In [11]:
```
# TASK 2

# Create a new column titled player status from the contract column with 3 labels:
      #  a. 'Active' if the player has an active contract
```

```
        #b. 'Free' if the player is free
       # c. 'On Loan' if the player is on loan
```

In [12]:
```
# checking for unique values on contract column
df.Contract.unique()
```

Out[12]:
```
array(['2004 ~ 2021', '2018 ~ 2022', '2014 ~ 2023', '2015 ~ 2023',
       '2017 ~ 2022', '2017 ~ 2023', '2018 ~ 2024', '2014 ~ 2022',
       '2018 ~ 2023', '2016 ~ 2023', '2013 ~ 2023', '2011 ~ 2023',
       '2009 ~ 2022', '2005 ~ 2021', '2011 ~ 2021', '2015 ~ 2022',
       '2017 ~ 2024', '2010 ~ 2024', '2012 ~ 2021', '2019 ~ 2024',
       '2015 ~ 2024', '2017 ~ 2025', '2020 ~ 2025', '2019 ~ 2023',
       '2008 ~ 2023', '2015 ~ 2021', '2020 ~ 2022', '2012 ~ 2022',
       '2016 ~ 2025', '2013 ~ 2022', '2011 ~ 2022', '2012 ~ 2024',
       '2016 ~ 2021', '2012 ~ 2023', '2008 ~ 2022', '2019 ~ 2022',
       '2017 ~ 2021', '2013 ~ 2024', '2020 ~ 2024', '2010 ~ 2022',
       '2020 ~ 2021', '2011 ~ 2024', '2020 ~ 2023', '2014 ~ 2024',
       '2013 ~ 2026', '2016 ~ 2022', '2010 ~ 2021', '2013 ~ 2021',
       '2019 ~ 2025', '2018 ~ 2025', '2016 ~ 2024', '2018 ~ 2021',
       '2009 ~ 2024', '2007 ~ 2022', 'Jun 30, 2021 On Loan',
       '2009 ~ 2021', '2019 ~ 2021', '2019 ~ 2026', 'Free', '2012 ~ 2028',
       '2010 ~ 2023', '2014 ~ 2021', '2015 ~ 2025', '2014 ~ 2026',
       '2012 ~ 2025', '2017 ~ 2020', '2002 ~ 2022', '2020 ~ 2027',
       '2013 ~ 2025', 'Dec 31, 2020 On Loan', '2019 ~ 2020',
       '2011 ~ 2025', '2016 ~ 2020', '2007 ~ 2021', '2020 ~ 2026',
       '2010 ~ 2025', '2009 ~ 2023', '2008 ~ 2021', '2020 ~ 2020',
       '2016 ~ 2026', 'Jan 30, 2021 On Loan', '2012 ~ 2020',
       '2014 ~ 2025', 'Jun 30, 2022 On Loan', '2015 ~ 2020',
       'May 31, 2021 On Loan', '2018 ~ 2020', '2014 ~ 2020',
       '2013 ~ 2020', '2006 ~ 2024', 'Jul 5, 2021 On Loan',
       'Dec 31, 2021 On Loan', '2004 ~ 2025', '2011 ~ 2020',
       'Jul 1, 2021 On Loan', 'Jan 1, 2021 On Loan', '2006 ~ 2023',
       'Aug 31, 2021 On Loan', '2006 ~ 2021', '2005 ~ 2023',
       '2003 ~ 2020', '2009 ~ 2020', '2002 ~ 2020', '2005 ~ 2020',
       '2005 ~ 2022', 'Jan 31, 2021 On Loan', '2010 ~ 2020',
       'Dec 30, 2021 On Loan', '2008 ~ 2020', '2007 ~ 2020',
       '2003 ~ 2021', 'Jun 23, 2021 On Loan', 'Jan 3, 2021 On Loan',
       'Nov 27, 2021 On Loan', '2002 ~ 2021', 'Jan 17, 2021 On Loan',
       'Jun 30, 2023 On Loan', '1998 ~ 2021', '2003 ~ 2022',
       '2007 ~ 2023', 'Jul 31, 2021 On Loan', 'Nov 22, 2020 On Loan',
       'May 31, 2022 On Loan', '2006 ~ 2020', 'Dec 30, 2020 On Loan',
       '2007 ~ 2025', 'Jan 4, 2021 On Loan', 'Nov 30, 2020 On Loan',
       '2004 ~ 2020', '2009 ~ 2025', 'Aug 1, 2021 On Loan'], dtype=object)
```

```
In [13]: contList = []


         for item in df.Contract.values:
             if 'free' in item:
                 contList.append('free')
             elif 'On Loan' in item:
                 contList.append('On Loan')
             else:
                 contList.append('Active')
```

```
In [14]: contList
```

```
Out[14]: ['Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'On Loan',
 'On Loan',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 'Active',
 ...]
```

In [15]:
```
# Adding playerstatus to my dataframe
df['playerStatus'] = contList
```

In [ ]:
```

```

In [16]:
```
# TASK 3
#   Unpack the POSITIONS column into as many columns as there are postions and assign boolean
#       values in the columns for each player as appropriate. Name the columns the play postion
```

In [17]:
```
df.Positions.unique()
```

```
Out[17]: array(['RW, ST, CF', 'ST, LW', 'GK', 'CAM, CM', 'LW, CAM', 'ST', 'RW',
                'ST, LW, RW', 'CB', 'LW', 'CDM', 'CF, ST', 'LW, RW', 'CDM, CM',
                'CDM, RB', 'CF, CAM', 'LW, ST', 'CM', 'ST, CF, LW', 'RM, LM, CAM',
                'RB', 'RW, CAM, CM', 'LB', 'LM, CF', 'CF', 'RW, LW', 'CAM, RM, RW',
                'CM, CDM', 'CAM, CF, ST', 'CM, CDM, CAM', 'CF, LW, CAM',
                'CAM, RM, CF', 'LM, ST', 'RM, LM, RW', 'LM', 'CAM, RW', 'CB, CDM',
                'RW, RM', 'LW, CF', 'CM, RM, LM', 'LB, LM', 'CAM, CM, RM',
                'CAM, CM, CF', 'CAM, CF', 'LM, RM, LW', 'LM, LB, CM', 'CM, LM, LB',
                'RM, RW', 'RM, CM', 'CAM, CM, LW', 'CB, LB', 'RM, RB', 'ST, RW',
                'LM, RW, LW', 'RB, LB', 'RB, RM', 'RM', 'LM, RM, CF', 'CAM, RM',
                'RB, RWB', 'CDM, CB, CM', 'CAM, RM, ST', 'LM, LW, RM', 'CM, CAM',
                'ST, RM, CF', 'LM, RM', 'RM, CF', 'LM, LWB', 'RW, RM, CF',
                'RB, CM', 'LW, CAM, RW', 'CAM, LW, CM', 'CM, CAM, CDM',
                'RW, LW, CAM', 'CM, CAM, LM', 'CM, RM, ST', 'CDM, CM, RB',
                'ST, CAM', 'CAM, LW, ST', 'LB, CB, LWB', 'RM, ST', 'CB, CDM, LB',
                'RWB, RM', 'CM, LM, RM', 'RB, CDM, CM', 'RW, LW, RM', 'LM, LW',
                'CM, LM', 'LM, LB', 'RM, LM, CF', 'LB, LM, RM', 'CDM, CM, CAM',
                'ST, LW, RM', 'CAM, CM, ST', 'ST, CF', 'LWB, LB', 'LW, RW, LM',
                'RM, RW, ST', 'LWB', 'CF, ST, CAM', 'LM, CAM, RM', 'RB, CB',
                'ST, LM', 'RW, CAM', 'LM, CAM', 'RWB, RB', 'ST, RW, LW', 'CAM',
                'RB, RM, RW', 'LB, LWB', 'RM, CAM', 'CAM, ST', 'CDM, CB',
                'CF, LM, LW', 'CAM, LM, LW', 'LW, RW, CAM', 'CB, RB',
                'RB, CB, RWB', 'LM, LW, ST', 'LW, ST, CM', 'RM, CAM, CM',
                'CB, RB, RWB', 'RW, ST, LW', 'LW, CAM, CM', 'CM, LM, CDM',
                'LB, LM, LWB', 'CB, LB, RB', 'CAM, LW', 'RWB, RB, RM',
                'CF, CAM, ST', 'RM, CAM, RW', 'RW, ST', 'LW, LM', 'RB, RM, CM',
                'ST, CAM, RW', 'CM, RB, LB', 'CAM, LM, RM', 'RB, RW', 'LM, CM',
                'RM, LM, CM', 'CDM, CM, CB', 'CM, CF', 'CF, LW, RW', 'ST, RM',
                'CAM, CM, CDM', 'LB, CB', 'RW, RWB', 'ST, LM, RM', 'RM, RWB, RB',
                'LM, ST, CAM', 'CAM, ST, CF', 'LW, CM', 'RB, RWB, LB', 'RM, LM',
                'CM, CAM, RM', 'ST, LW, CAM', 'RM, RW, CAM', 'CM, CDM, RM',
                'RB, RM, RWB', 'CDM, CB, LB', 'CAM, ST, LM', 'LM, CM, RM',
                'CF, RW', 'CAM, RM, LW', 'CM, RM, CDM', 'LB, LWB, LM',
                'LW, RW, CF', 'LW, LM, CAM', 'LWB, LW, LB', 'CDM, CM, LM',
                'CB, CM', 'RWB, RW, RB', 'ST, RW, RM', 'LW, RW, CM', 'LM, RM, CAM',
                'LB, RB', 'RWB, RM, RB', 'LM, RB, LB', 'LW, LM, ST', 'ST, RM, LM',
                'CAM, CM, LM', 'CF, CM', 'LW, ST, RW', 'ST, CAM, CF', 'LB, CB, LM',
                'CM, CDM, LM', 'RM, RWB', 'LB, LM, RB', 'RB, LB, CB',
                'CM, CDM, RB', 'LB, CM', 'CF, CAM, LW', 'RM, LWB', 'CF, ST, LM',
                'LB, RB, CB', 'LW, CF, ST', 'RM, ST, CAM', 'LW, RW, ST',
                'ST, LW, LM', 'CM, LW', 'CDM, CM, LB', 'CM, RM, CAM',
                'ST, CF, CAM', 'CM, LM, CAM', 'RWB', 'LM, ST, RM', 'CAM, RM, CM',
                'CAM, LM, CM', 'RW, LW, ST', 'CAM, RM, LM', 'CF, ST, LW',
                'LWB, LB, LM', 'RM, CM, CAM', 'LB, LM, CAM', 'CAM, CDM',
                'RW, RB, RM', 'RM, ST, RW', 'CM, CAM, LW', 'CF, ST, RW',
                'LM, RM, RB', 'LM, RM, CM', 'CB, RB, RM', 'RB, CDM', 'RM, ST, LM',
                'LB, LWB, CB', 'CDM, LB', 'LM, RM, ST', 'RWB, RB, LWB',
                'ST, LM, CAM', 'ST, RM, CAM', 'RB, RWB, RM', 'CF, LM, CAM',
                'CAM, CF, RW', 'RB, RM, LB', 'CDM, RWB', 'CM, RW, CAM',
                'ST, CF, LM', 'RM, LM, RWB', 'RB, LB, RM', 'LM, LW, LB',
                'RM, LM, ST', 'CAM, LM', 'ST, LM, LW', 'LW, RW, RM', 'CF, LW',
                'LB, CM, RB', 'RB, CB, CDM', 'CB, RB, LB', 'CAM, RW, RM',
                'LWB, LM', 'LW, CF, LM', 'CAM, CF, CM', 'LWB, CB', 'CM, CDM, LW',
                'LM, CM, LB', 'CM, RM', 'RW, LW, CF', 'CM, CDM, CB', 'LM, RM, RW',
                'RM, RW, LM', 'RM, CAM, LM', 'LWB, RM, LB', 'RM, LB, RB',
                'CM, CF, RM', 'RM, RB, LB', 'ST, LM, RW', 'CAM, ST, LW',
                'CF, LW, ST', 'LM, LW, CAM', 'RM, RW, CF', 'LW, LM, CF',
                'CM, RW, LW', 'LM, CM, CAM', 'CAM, RW, ST', 'CM, CAM, CF',
                'RW, LWB, LW', 'CB, RB, CDM', 'RW, RM, LW', 'LW, ST, LM',
                'RWB, RW', 'ST, CAM, LM', 'CM, CB', 'RM, RW, CM', 'LWB, CM',
                'RM, LM, LB', 'RM, CAM, ST', 'CDM, RM, RB', 'LM, LB, RB',
                'LB, RB, RWB', 'RM, RWB, CAM', 'CAM, LM, CF', 'LM, CAM, CM',
                'CF, CAM, RW', 'CDM, CM, RM', 'CF, CAM, CM', 'ST, RM, LW',
                'CB, CDM, CM', 'RB, RW, LB', 'ST, RW, CAM', 'CM, LB', 'LW, RM, RW',
                'CM, RM, RW', 'RM, CF, LM', 'CF, LM, RM', 'CAM, ST, RM', 'RW, CF',
                'CM, CAM, ST', 'CAM, CDM, CM', 'RM, RW, LW', 'CAM, LM, LB',
                'CAM, RW, LW', 'CDM, CAM', 'LWB, LM, LB', 'RW, LB', 'LW, CAM, CF',
                'RB, RWB, CB', 'LM, CF, RM', 'RB, LB, RWB', 'RM, LW',
                'CAM, LM, ST', 'LW, LM, LB', 'LB, RB, RM', 'CM, LWB',
                'CDM, RB, RM', 'RM, CM, RB', 'ST, RM, RW', 'LM, RM, LWB',
                'RW, LW, LM', 'LW, CF, RW', 'CM, CDM, LB', 'ST, LM, RB', 'LB, CDM',
                'CM, CB, CDM', 'CB, LB, CDM', 'RW, ST, RM', 'RM, ST, CF',
                'CAM, LW, LM', 'LM, ST, LW', 'CAM, RM, RB', 'RB, CM, RM',
                'CM, LW, LM', 'ST, CAM, RM', 'RW, ST, CAM', 'CM, LB, LM',
                'RB, CDM, RM', 'LM, RWB, RM', 'LM, CAM, LB', 'CAM, LW, RW',
                'CM, RM, RWB', 'RW, CAM, LW', 'RB, CB, LB', 'LB, LM, LW', 'RW, RB',
                'LM, LWB, CM', 'RM, CAM, RB', 'RM, CM, ST', 'RB, RM, LM',
                'LM, LB, LW', 'LM, ST, CF', 'RW, LWB, RM', 'LM, CAM, LW',
                'CB, RWB', 'RM, LW, CAM', 'LB, LW', 'CDM, CM, ST', 'LB, CM, CDM',
                'CF, CAM, LM', 'RW, CAM, RM', 'RW, LM, LW', 'RW, CM, CAM',
                'LM, LW, RW', 'RM, LW, RW', 'LW, LWB, RWB', 'RM, LM, RB',
                'CB, CDM, RB', 'CAM, RW, CM', 'ST, CAM, CM', 'LWB, LM, CM',
                'RW, LW, RWB', 'CB, LB, LWB', 'CM, LM, ST', 'CM, LWB, LM',
                'LB, LW, LM', 'RB, CM, CAM', 'RM, CAM, CF', 'CM, RW',
                'RW, RM, CAM', 'CB, LWB', 'CAM, RB', 'LM, CM, CDM', 'RM, LB, LM',
                'CB, RB, CM', 'RB, CM, RWB', 'LB, RB, LM', 'CAM, ST, RW',
                'LB, LW, RW', 'CDM, CAM, CM', 'CAM, ST, CM', 'LM, CAM, ST',
                'LM, ST, CM', 'CAM, CF, RM', 'CF, ST, RM', 'LM, ST, LB',
                'ST, CAM, LW', 'LWB, CM, LB', 'RWB, RM, LWB', 'LWB, LM, RB',
                'CM, LW, ST', 'LB, LM, CB', 'CM, RM, RB', 'RWB, CB', 'RM, RB, RWB',
                'LB, CM, LM', 'ST, LW, CF', 'RM, RWB, ST', 'CF, ST, CM',
                'LM, LB, LWB', 'LM, RW', 'LM, LWB, LB', 'LB, LWB, RB',
                'RM, RB, LM', 'RM, CM, LM', 'LW, ST, CAM', 'RB, RM, CB',
                'LWB, RWB, LB', 'CF, RM, RW', 'RM, CF, CAM', 'LM, LW, CF',
                'RB, LM, LB', 'CAM, CM, RW', 'LB, CM, LW', 'CM, CF, CAM',
                'RW, RM, ST', 'CDM, RB, CM', 'CM, RB', 'RB, RWB, RW', 'LM, RW, RM',
                'CM, RB, CDM', 'LM, RM, LB', 'CDM, RB, CB', 'RB, CB, RM',
                'RWB, LW, LM', 'ST, CF, RW', 'RM, CM, CDM', 'LM, CF, ST',
                'LW, CAM, LM', 'CDM, LM, CM', 'ST, CM, CAM', 'LM, RM, RWB',
                'CM, CAM, RW', 'CAM, LM, RW', 'LW, LM, RM', 'CAM, RW, CF',
                'LM, ST, RW', 'CB, LWB, LB', 'RW, CF, LW', 'CB, CM, CDM',
                'CB, CAM', 'LW, LB', 'CDM, RB, LB', 'ST, CM, RB', 'RWB, LWB, RB',
                'CM, CB, RB', 'LB, LW, RB', 'CF, RM, RWB', 'LB, RM', 'RM, RW, RB',
                'LB, RM, RB', 'CDM, RM, CM', 'CDM, LB, CM', 'LM, CDM, CM',
                'RW, LM, CAM', 'LM, LB, RM', 'RM, CF, ST', 'RW, CM', 'RM, RWB, LM',
                'CF, CM, ST', 'RW, RM, LM', 'LM, CM, RW', 'RWB, CM', 'RB, RM, CDM',
                'CM, CB, CAM', 'CF, CM, LM', 'RWB, LWB, RM', 'CF, RM',
                'CM, CF, ST', 'RWB, RB, CB', 'LWB, RWB', 'RM, LM, LW',
                'LW, RM, CM', 'LW, CAM, ST', 'ST, RM, RWB', 'ST, CM',
                'CDM, CM, RWB', 'LB, CB, RB', 'CB, ST, CAM', 'CF, RM, CM',
                'CDM, CB, RB', 'CM, LM, RB', 'RB, CB, LWB', 'LM, LB, CDM',
                'LW, LWB', 'RW, CF, CAM', 'RM, RB, CAM', 'LB, RB, CM',
                'RM, CDM, LM', 'LWB, LB, CB', 'LB, RB, CDM', 'RB, CM, LB',
                'CDM, RM', 'LM, RB, CB', 'RW, RWB, RM', 'RM, LW, ST',
                'LB, RB, LWB', 'RM, LM, CDM', 'RB, RM, CAM', 'RM, ST, LW',
                'CF, LM', 'RB, LB, CDM', 'RW, LW, RB', 'RB, RWB, CDM',
                'CM, CAM, LB', 'LW, RW, LWB', 'CB, LWB, RWB', 'LB, CB, CDM',
                'ST, RB, RM', 'CM, RB, RM', 'RW, LM', 'RM, CF, RB', 'CB, RM, RB',
                'RWB, LWB', 'CM, ST, CDM', 'LWB, LW', 'CB, RB, ST', 'LB, LWB, CDM',
                'CB, RWB, RM', 'CB, ST', 'CM, ST', 'LM, LWB, CB', 'ST, LW, LWB',
                'LM, CDM, RM', 'LWB, LB, RW', 'RB, CM, CB', 'LM, RW, ST',
                'CM, RM, LW', 'ST, RW, LM', 'RWB, CM, RM', 'CAM, RWB, CM',
                'CM, RB, LM', 'RM, RB, RW', 'LW, LM, CM', 'RM, LM, LWB',
                'ST, LM, CF', 'LW, RM, CAM', 'RB, RWB, LWB', 'LB, CM, ST',
                'ST, LWB', 'RB, ST, LM', 'CDM, CAM, LM', 'LM, ST, LWB',
                'LB, RB, RW', 'RM, CDM', 'RB, LB, LWB', 'RW, CAM, LM',
                'LB, CDM, LM', 'RW, RB, LM', 'CB, LM', 'RW, LM, RM', 'RM, LB, CM',
                'LW, CAM, RM', 'RW, RB, LB', 'LW, LB, LM', 'CDM, RB, RWB',
                'CB, RW', 'CM, LM, LW', 'LM, RM, CB', 'LM, LWB, ST', 'ST, CB, CDM',
                'LB, CDM, CB', 'CDM, CB, RM', 'RB, RWB, CM', 'ST, RW, CF',
                'ST, CB', 'ST, LW, CM', 'LM, LB, ST', 'ST, CB, RB', 'RB, LM',
                'GK, RB', 'LM, LW, CM', 'LM, CM, LW', 'CB, CAM, CM', 'LM, RB',
                'RWB, CAM', 'RB, CAM', 'CB, CDM, RM', 'ST, RB', 'ST, RWB',
                'CAM, CDM, LM', 'CB, RWB, RB', 'LM, LWB, RM', 'LB, LM, CM',
                'CM, CF, LM', 'CM, RWB', 'CDM, LM', 'ST, CF, RM', 'CAM, LB',
                'RB, RM, ST', 'LM, CDM', 'CDM, RW, RB', 'LM, CF, CAM',
                'LWB, LM, RWB', 'CF, RM, ST', 'CAM, CF, LM', 'RB, ST, CB',
                'RW, RB, LW', 'LB, LM, CDM', 'RB, CB, ST', 'RWB, CB, RB',
                'CDM, ST', 'LW, RM, LM', 'RB, LWB', 'CDM, LB, CB', 'LM, LB, CF',
                'RB, CDM, LB', 'LB, LWB, CM', 'RM, LM, CB', 'CAM, LW, CF',
                'LB, LW, CM', 'RB, CM, CDM', 'LWB, LM, ST', 'CM, RW, RM', 'CB, RM',
                'CM, LM, CB', 'LM, LB, CB', 'ST, RW, CM', 'RM, ST, CM',
                'RWB, RM, LM', 'CM, CAM, RB', 'CM, RWB, CDM', 'LB, LWB, LW'],
               dtype=object)
```

In [18]:
```
posList = []

for val in df.Positions:
    if ',' in val:
        pList = val.split(',')
        playPos = [x.strip() for x in pList]
        posList.extend(playPos)

    else:
        val.strip()
        posList.append(val)
```

In [19]:
```
pos = list(set(posList))
```

In [20]:
```
pos
```

Out[20]:
```
['CDM',
 'LM',
 'ST',
 'CAM',
 'CM',
 'LB',
 'RWB',
 'RB',
 'LWB',
 'CF',
 'LW',
 'RW',
 'GK',
 'RM',
 'CB']
```

In [21]:
```
playerPos = []

for Positions in df.Positions.values:
    posTable = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,]
    for p in pos:
        if p in Positions:
            pidx = pos.index(p)
            posTable[pidx] = 1
    playerPos.append(posTable)
```

In [22]:
```
playerPos
```

```
Out[22]: [[0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
          [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
          [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0],
          [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
          ...]
```

In [23]:
```
# converting the positions list to a dataframe
dfpos = pd.DataFrame(playerPos,columns = pos)
```

In [24]:
```
dfpos
```

Out[24]:

	CDM	LM	ST	CAM	CM	LB	RWB	RB	LWB	CF	LW	RW	GK	RM	CB
0	0	0	1	0	0	0	0	0	0	1	0	1	0	0	0
1	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
3	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
19016	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
19017	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
19018	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
19019	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
19020	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1

19021 rows × 15 columns

In [25]:
```
df.Positions.values[0:5]
```

```
Out[25]: array(['RW, ST, CF', 'ST, LW', 'GK', 'CAM, CM', 'LW, CAM'], dtype=object)
```

```
In [26]: # Preserving a copy of our original data
         dfNew = df.copy()
```

```
In [ ]:
```

```
In [27]: # adding position to my new dataframe. dfnew
         dfNew = df.join(dfpos)
```

```
In [28]: dfNew
```

Out[28]:

	ID	Name	LongName	photoUrl	playerUrl	Nationality	Age
0	158023	L. Messi	Lionel Messi	https://cdn.sofifa.com/players/158/023/21_60.png	http://sofifa.com/player/158023/lionel-messi/2...	Argentina	33
1	20801	Cristiano Ronaldo	C. Ronaldo dos Santos Aveiro	https://cdn.sofifa.com/players/020/801/21_60.png	http://sofifa.com/player/20801/c-ronaldo-dos-s...	Portugal	35
2	200389	J. Oblak	Jan Oblak	https://cdn.sofifa.com/players/200/389/21_60.png	http://sofifa.com/player/200389/jan-oblak/210006/	Slovenia	27
3	192985	K. De Bruyne	Kevin De Bruyne	https://cdn.sofifa.com/players/192/985/21_60.png	http://sofifa.com/player/192985/kevin-de-bruyn...	Belgium	29
4	190871	Neymar Jr	Neymar da Silva Santos Jr.	https://cdn.sofifa.com/players/190/871/21_60.png	http://sofifa.com/player/190871/neymar-da-silv...	Brazil	28
...	...	...	...	...	...	...	...
19016	247223	Xia Ao	Ao Xia	https://cdn.sofifa.com/players/247/223/21_60.png	http://sofifa.com/player/247223/ao-xia/210006/	China PR	21
19017	258760	B. Hough	Ben Hough	https://cdn.sofifa.com/players/258/760/21_60.png	http://sofifa.com/player/258760/ben-hough/210006/	England	17
19018	252757	R. McKinley	Ronan McKinley	https://cdn.sofifa.com/players/252/757/21_60.png	http://sofifa.com/player/252757/ronan-mckinley...	England	18
19019	243790	Wang Zhen'ao	Zhen'ao Wang	https://cdn.sofifa.com/players/243/790/21_60.png	http://sofifa.com/player/243790/zhenao-wang/21...	China PR	20
19020	252520	Zhou Xiao	Xiao Zhou	https://cdn.sofifa.com/players/252/520/21_60.png	http://sofifa.com/player/252520/xiao-zhou/210006/	China PR	21

19021 rows × 94 columns

In [ ]:

In [29]:

```
# TASK 4
#   Weight and Height, W/F, SM AND IR columns: convert to intergers
```

```
In [30]: # checking for unique value in weight
         dfNew.Weight.unique()

Out[30]: array(['72kg', '83kg', '87kg', '70kg', '68kg', '80kg', '71kg', '91kg',
                '73kg', '85kg', '92kg', '69kg', '84kg', '96kg', '81kg', '82kg',
                '75kg', '86kg', '89kg', '74kg', '76kg', '64kg', '78kg', '90kg',
                '66kg', '60kg', '94kg', '79kg', '67kg', '65kg', '59kg', '61kg',
                '93kg', '88kg', '97kg', '77kg', '62kg', '63kg', '95kg', '100kg',
                nan, '58kg', '183lbs', '179lbs', '172lbs', '196lbs', '176lbs',
                '185lbs', '170lbs', '203lbs', '168lbs', '161lbs', '146lbs',
                '130lbs', '190lbs', '174lbs', '148lbs', '165lbs', '159lbs',
                '192lbs', '181lbs', '139lbs', '154lbs', '157lbs', '163lbs', '98kg',
                '103kg', '99kg', '102kg', '56kg', '101kg', '57kg', '55kg', '104kg',
                '107kg', '110kg', '53kg', '50kg', '54kg', '52kg'], dtype=object)
```

```
In [31]: # Converting the Weight feature to int

         wList = []

         for val in df['Weight']:
             if pd.isna(val): # incase there is a missing values
                 wList.append(val)
             else:
                 val = str(val) # Converting all values in the Weight feature to string
                 if 'kg' not in val: # Converting (lbs) to kg
                     lbs = int(val[:-3]) * 0.45359
                     wList.append(lbs)
                 else:
                     kg  = int(val[:-2])
                     wList.append(kg)
         dfNew['weight'] = wList
```

```
In [ ]:
```

```
In [32]: # checking for unique values in height column.

         df.Height.unique()
```

```
Out[32]: array(['170cm', '187cm', '188cm', '181cm', '175cm', '184cm', '191cm',
                '178cm', '193cm', '185cm', '199cm', '173cm', '168cm', '176cm',
                '177cm', '183cm', '180cm', '189cm', '179cm', '195cm', '172cm',
                '182cm', '186cm', '192cm', '165cm', '194cm', '167cm', '196cm',
                '163cm', '190cm', '174cm', '169cm', '171cm', '197cm', '200cm',
                '166cm', '6\'2"', '164cm', '198cm', '6\'3"', '6\'5"', '5\'11"',
                '6\'4"', '6\'1"', '6\'0"', '5\'10"', '5\'9"', '5\'6"', '5\'7"',
                '5\'4"', '201cm', '158cm', '162cm', '161cm', '160cm', '203cm',
                '157cm', '156cm', '202cm', '159cm', '206cm', '155cm'], dtype=object)
```

```
In [33]: tempList = []

         for val in df.Height.values:
             if 'cm' not in val:
                 tempList.append(val)

         xx = list(set(tempList))
         print(xx)
```

```
['5\'6"', '6\'4"', '5\'7"', '6\'2"', '6\'3"', '5\'10"', '6\'0"', '5\'9"', '6\'5"', '5\'4"', '5\'11"', '6\'1"']
```

```
In [34]: # Converting the Height feature to int
         hgtList = []

         for val in df.Height.values:
             val = str(val)
             if 'cm' in val:
                 ht = int(val[:-2])
                 hgtList.append(ht)
             elif val in xx:
                 ft = val[0]
                 inch = val[-2]
                 hgt1 = int(ft)*30.48 + int(inch)*2.54
                 hgtList.append(hgt1)
             else:
                 hgtList.append(val)
         dfNew['height'] = hgtList
```

```
In [ ]:
```

```
In [35]: # Converting W/F features to int
         wf = []
         for val in df['W/F']:
             val = int(val[0][0])
```

```
    wf.append(val)
dfNew['W/F1'] = wf
```

In [ ]:

```
```

In [36]:

```
# Converting SM features to int
sm = []
for val in df['SM']:
    val = int(val[0][0])
    sm.append(val)
dfNew['SM1'] = sm
```

In [ ]:

```
```

In [37]:

```
# Converting SM features to int
ir = []
for val in df['IR']:
    val = int(val[0][0])
    ir.append(val)
dfNew['IR1'] = ir
```

In [38]:

```
# inspecting my dataframe
dfNew
```

Out[38]:

	ID	Name	LongName	photoUrl	playerUrl	Nationality	Age
0	158023	L. Messi	Lionel Messi	https://cdn.sofifa.com/players/158/023/21_60.png	http://sofifa.com/player/158023/lionel-messi/2...	Argentina	33
1	20801	Cristiano Ronaldo	C. Ronaldo dos Santos Aveiro	https://cdn.sofifa.com/players/020/801/21_60.png	http://sofifa.com/player/20801/c-ronaldo-dos-s...	Portugal	35
2	200389	J. Oblak	Jan Oblak	https://cdn.sofifa.com/players/200/389/21_60.png	http://sofifa.com/player/200389/jan-oblak/210006/	Slovenia	27
3	192985	K. De Bruyne	Kevin De Bruyne	https://cdn.sofifa.com/players/192/985/21_60.png	http://sofifa.com/player/192985/kevin-de-bruyn...	Belgium	29
4	190871	Neymar Jr	Neymar da Silva Santos Jr.	https://cdn.sofifa.com/players/190/871/21_60.png	http://sofifa.com/player/190871/neymar-da-silv...	Brazil	28
...	...	...	...	...	...	...	...
19016	247223	Xia Ao	Ao Xia	https://cdn.sofifa.com/players/247/223/21_60.png	http://sofifa.com/player/247223/ao-xia/210006/	China PR	21
19017	258760	B. Hough	Ben Hough	https://cdn.sofifa.com/players/258/760/21_60.png	http://sofifa.com/player/258760/ben-hough/210006/	England	17
19018	252757	R. McKinley	Ronan McKinley	https://cdn.sofifa.com/players/252/757/21_60.png	http://sofifa.com/player/252757/ronan-mckinley...	England	18
19019	243790	Wang Zhen'ao	Zhen'ao Wang	https://cdn.sofifa.com/players/243/790/21_60.png	http://sofifa.com/player/243790/zhenao-wang/21...	China PR	20
19020	252520	Zhou Xiao	Xiao Zhou	https://cdn.sofifa.com/players/252/520/21_60.png	http://sofifa.com/player/252520/xiao-zhou/210006/	China PR	21

19021 rows × 99 columns

In [ ]:

```
```

In [39]:

```
# TASK 5

#   Value, Wage and Release Clause columns: convert to float
```

```
In [40]: # wage unique values
         dfNew.Wage.unique()

Out[40]: array(['€560K', '€220K', '€125K', '€370K', '€270K', '€240K', '€250K',
                '€160K', '€260K', '€210K', '€310K', '€130K', '€350K', '€300K',
                '€190K', '€145K', '€195K', '€100K', '€140K', '€290K', '€82K',
                '€110K', '€230K', '€155K', '€200K', '€165K', '€95K', '€170K',
                '€105K', '€115K', '€150K', '€135K', '€55K', '€58K', '€81K', '€34K',
                '€120K', '€59K', '€90K', '€65K', '€56K', '€71K', '€18K', '€75K',
                '€47K', '€20K', '€84K', '€86K', '€74K', '€78K', '€27K', '€68K',
                '€85K', '€25K', '€46K', '€83K', '€54K', '€79K', '€175K', '€43K',
                '€49K', '€45K', '€38K', '€41K', '€39K', '€23K', '€51K', '€50K',
                '€87K', '€30K', '€14K', '€69K', '€31K', '€64K', '€53K', '€35K',
                '€21K', '€28K', '€17K', '€33K', '€70K', '€32K', '€89K', '€26K',
                '€40K', '€76K', '€72K', '€48K', '€36K', '€29K', '€60K', '€16K',
                '€37K', '€24K', '€52K', '€0', '€62K', '€73K', '€63K', '€19K',
                '€1K', '€66K', '€80K', '€12K', '€2K', '€42K', '€13K', '€900',
                '€57K', '€77K', '€61K', '€22K', '€67K', '€44K', '€15K', '€11K',
                '€8K', '€850', '€10K', '€88K', '€500', '€7K', '€6K', '€9K', '€5K',
                '€700', '€950', '€750', '€3K', '€650', '€600', '€4K', '€800',
                '€550'], dtype=object)

In [41]: # Converting the Wage feature to float
         wageList = []
         for val in df['Wage']:
             if 'K' in val:
                 val = float(val[1:-1]) * 1000
                 wageList.append(val)
             else:
                 val =  float(val[1:]) * 1000
                 wageList.append(val)
         dfNew['wage'] = wageList

In [ ]:

In [42]: dfNew.Value.unique()
```

```
Out[42]: array(['€103.5M', '€63M', '€120M', '€129M', '€132M', '€111M', '€120.5M',
                '€102M', '€185.5M', '€110M', '€113M', '€90.5M', '€82M', '€17.5M',
                '€83.5M', '€33.5M', '€114.5M', '€78M', '€103M', '€109M', '€92M',
                '€10M', '€76.5M', '€89.5M', '€87.5M', '€79.5M', '€124M', '€114M',
                '€95M', '€92.5M', '€105.5M', '€88.5M', '€85M', '€81.5M', '€26M',
                '€21M', '€56M', '€67.5M', '€53M', '€36.5M', '€51M', '€65.5M',
                '€46.5M', '€61.5M', '€72.5M', '€77.5M', '€43.5M', '€32.5M', '€36M',
                '€32M', '€54M', '€49.5M', '€57M', '€66.5M', '€74.5M', '€71.5M',
                '€121M', '€99M', '€67M', '€86.5M', '€93.5M', '€70M', '€62M',
                '€66M', '€58M', '€44M', '€81M', '€37M', '€14.5M', '€46M', '€47.5M',
                '€52.5M', '€54.5M', '€34.5M', '€57.5M', '€51.5M', '€44.5M', '€55M',
                '€48M', '€60.5M', '€63.5M', '€61M', '€29M', '€58.5M', '€55.5M',
                '€42M', '€40.5M', '€43M', '€45.5M', '€34M', '€26.5M', '€42.5M',
                '€35.5M', '€45M', '€41.5M', '€40M', '€11M', '€13.5M', '€29.5M',
                '€27M', '€15.5M', '€38.5M', '€52M', '€33M', '€19M', '€73.5M',
                '€38M', '€35M', '€47M', '€24M', '€30.5M', '€18M', '€28M', '€25.5M',
                '€25M', '€31M', '€23.5M', '€30M', '€31.5M', '€22.5M', '€28.5M',
                '€4M', '€12.5M', '€37.5M', '€27.5M', '€16M', '€15M', '€20.5M',
                '€22M', '€3.4M', '€5M', '€56.5M', '€62.5M', '€0', '€39M', '€24.5M',
                '€21.5M', '€13M', '€8M', '€20M', '€8.5M', '€2.9M', '€9M', '€4.6M',
                '€50M', '€23M', '€18.5M', '€7M', '€19.5M', '€5.5M', '€7.5M',
                '€3.8M', '€14M', '€10.5M', '€16.5M', '€3.6M', '€9.5M', '€39.5M',
                '€17M', '€12M', '€11.5M', '€4.9M', '€3M', '€1.9M', '€6.5M',
                '€1.7M', '€2.4M', '€3.1M', '€6M', '€3.7M', '€4.7M', '€4.3M',
                '€2.1M', '€1.2M', '€1.8M', '€4.8M', '€3.2M', '€1.3M', '€825K',
                '€2.3M', '€1.5M', '€3.9M', '€2.6M', '€3.5M', '€2.8M', '€2.7M',
                '€4.4M', '€4.1M', '€950K', '€1.6M', '€625K', '€1.1M', '€4.5M',
                '€4.2M', '€2.2M', '€3.3M', '€1.4M', '€2M', '€475K', '€925K',
                '€750K', '€725K', '€2.5M', '€1M', '€350K', '€525K', '€600K',
                '€850K', '€800K', '€550K', '€250K', '€400K', '€425K', '€575K',
                '€210K', '€325K', '€900K', '€875K', '€650K', '€700K', '€500K',
                '€975K', '€375K', '€775K', '€275K', '€180K', '€450K', '€675K',
                '€150K', '€240K', '€300K', '€130K', '€220K', '€200K', '€110K',
                '€170K', '€230K', '€90K', '€120K', '€80K', '€190K', '€140K',
                '€160K', '€100K', '€60K', '€50K', '€70K', '€45K', '€35K', '€40K',
                '€25K', '€20K', '€15K', '€30K', '€9K'], dtype=object)
```

```
In [ ]:
```

```
In [43]: # Converting the value feature to float
         valList = []
         for val in df['Value']:
             if 'M' in val:
                 mil = float(val[1:-1]) * 1000000
                 valList.append(mil)
```

```
    elif 'K' in val:
        thsd = float(val[1:-1]) * 1000
        valList.append(thsd)
    else:# Running a check to see if there are values without (M or K)
        val = float(val[1:])
        valList.append(val)
dfNew['value'] = valList
```

In [ ]:

In [44]:
```
# Converting the Release Clause feature to float
rvalList = []
for val in df['Release Clause']:
    if pd.isna(val):
        rvalList.append(val)
    else:
        if 'M' in val:
            mil = float(val[1:-1]) * 1000000
            rvalList.append(mil)
        elif 'K' in val:
            thsd = float(val[1:-1]) * 1000
            rvalList.append(thsd)
        else:
            val = float(val[1:])
            rvalList.append(val)
dfNew['Release clause'] = rvalList
```

In [ ]:

In [45]:
```
# Converting The Hits Feature From Object To Float
hitList = []
for val in df['Hits']:
    if pd.isna(val):
        hitList.append(val)
    else:
        val = str(val)
        if 'K' in val:
            thsd = float(val[:-1]) * 1000
            hitList.append(thsd)
        else:
            val = float(val)
            hitList.append(val)
dfNew['HITS'] = hitList
```

```
In [ ]:
```

```
In [46]: # inspecting my dataframe
         dfNew
```

Out[46]:

	ID	Name	LongName	photoUrl	playerUrl	Nationality	Age
0	158023	L. Messi	Lionel Messi	https://cdn.sofifa.com/players/158/023/21_60.png	http://sofifa.com/player/158023/lionel-messi/2...	Argentina	33
1	20801	Cristiano Ronaldo	C. Ronaldo dos Santos Aveiro	https://cdn.sofifa.com/players/020/801/21_60.png	http://sofifa.com/player/20801/c-ronaldo-dos-s...	Portugal	35
2	200389	J. Oblak	Jan Oblak	https://cdn.sofifa.com/players/200/389/21_60.png	http://sofifa.com/player/200389/jan-oblak/210006/	Slovenia	27
3	192985	K. De Bruyne	Kevin De Bruyne	https://cdn.sofifa.com/players/192/985/21_60.png	http://sofifa.com/player/192985/kevin-de-bruyn...	Belgium	29
4	190871	Neymar Jr	Neymar da Silva Santos Jr.	https://cdn.sofifa.com/players/190/871/21_60.png	http://sofifa.com/player/190871/neymar-da-silv...	Brazil	28
...	...	...	...	...	...	...	...
19016	247223	Xia Ao	Ao Xia	https://cdn.sofifa.com/players/247/223/21_60.png	http://sofifa.com/player/247223/ao-xia/210006/	China PR	21
19017	258760	B. Hough	Ben Hough	https://cdn.sofifa.com/players/258/760/21_60.png	http://sofifa.com/player/258760/ben-hough/210006/	England	17
19018	252757	R. McKinley	Ronan McKinley	https://cdn.sofifa.com/players/252/757/21_60.png	http://sofifa.com/player/252757/ronan-mckinley...	England	18
19019	243790	Wang Zhen'ao	Zhen'ao Wang	https://cdn.sofifa.com/players/243/790/21_60.png	http://sofifa.com/player/243790/zhenao-wang/21...	China PR	20
19020	252520	Zhou Xiao	Xiao Zhou	https://cdn.sofifa.com/players/252/520/21_60.png	http://sofifa.com/player/252520/xiao-zhou/210006/	China PR	21

19021 rows × 103 columns

```
In [ ]:
```

```
In [47]: # TASK 7
         # create 5 new categorical columns for the HEIGHT, WEIGHT, RELEASE CLAUSE, VALUE AND WAGE into
         #       which you convert the respective values into clusters/labels as follows
         #        a. Height: Bucket intervals of 10CM
         #        b. weight: Bucket intervals of 10 kg
         #        c. Wage: Bucket intervals of 50M
         #        d. value: bucket intervals of 50m
         #        e. Release Clause: bucket intervals of 50m
```

```
In [48]: #  (a) Height: Bucket intervals of 10CM


         min(dfNew.height)

Out[48]: 152.4
```

```
In [49]: max(dfNew['height'])

Out[49]: 206.0
```

```
In [50]: # Creating a categorical column for height in a bucket of 10cm
         upperbands = []
         last_bin = 210 # Creating a bin to hold values greater than 200
         counts = 1
         while counts <= max(dfNew['height'])/10:
             upperbands.append(counts * 10)
             counts +=1
         if last_bin not in upperbands:
             upperbands.append(last_bin)
         dfNew['height_bins'] = pd.cut(x = dfNew['height'],bins = upperbands)
```

```
In [ ]:
```

```
In [51]: #   b. weight: Bucket intervals of 10kg

         max(dfNew.weight)

Out[51]: 110.0
```

```
In [52]: min(dfNew.weight)

Out[52]: 50.0
```

```
In [53]: # Creating a categorical column for Weight in a bucket of 10kg
         upperbands = []
         counts = 1
         while counts <= max(dfNew['weight'])/10:
             upperbands.append(counts * 10)
             counts +=1
         dfNew['weight_bins'] = pd.cut(x = dfNew['weight'],bins = upperbands)
```

```
In [ ]:
```

```
In [54]: # c. Wage: Bucket intervals of 50k
         max(dfNew.wage)

Out[54]: 950000.0
```

```
In [55]: min(dfNew.wage)

Out[55]: 0.0
```

```
In [56]: # Creating a categorical column for Wage in a bucket of 50K
         upperbands = [0]
         counts = 1
         stop_bin = 960000
         while counts <= max(dfNew['wage'])/50000:
             upperbands.append(counts * 50000)
             counts +=1
         if stop_bin not in upperbands:
             upperbands.append(stop_bin)
         dfNew['wage_bins'] = pd.cut(x = dfNew['wage'],bins = upperbands,right = False)
```

```
In [ ]:
```

```
In [57]: # d. value: bucket intervals of 50m

         max(dfNew.value)

Out[57]: 185500000.0
```

```
In [58]: min(dfNew.value)

Out[58]: 0.0
```

```
In [59]: # Creating a categorical column for Value in a bucket of 50M
         upperbands = [0]
         counts = 1
         stop_bin = 200000000
         while counts <= max(dfNew['value'])/50000000:
             upperbands.append(counts * 50000000)
             counts +=1
         if stop_bin not in upperbands:
             upperbands.append(stop_bin)
         dfNew['value_bins'] = pd.cut(x = dfNew['value'],bins = upperbands,right = False)
```

```
In [ ]:
```

```
In [60]: max(dfNew['Release clause'])

Out[60]: 203100000.0
```

```
In [61]: min(dfNew['Release clause'])

Out[61]: 0.0
```

```
In [62]: # Creating a categorical column for Release Clause in a bucket of 50M
         upperbands = [0]
         counts = 1
         stop_bin = 250000000
         while counts <= max(dfNew['Release clause'])/50000000:
             upperbands.append(counts * 50000000)
             counts +=1
         if stop_bin not in upperbands:
             upperbands.append(stop_bin)
         dfNew['Release clause_bins'] = pd.cut(x = dfNew['Release clause'],bins = upperbands,right = False)
```

```
In [ ]:
```

```
In [63]: # INSPECTING MY DATAFRAME
         dfNew
```

Out[63]:

	ID	Name	LongName	photoUrl	playerUrl	Nationality	Age
0	158023	L. Messi	Lionel Messi	https://cdn.sofifa.com/players/158/023/21_60.png	http://sofifa.com/player/158023/lionel-messi/2...	Argentina	33
1	20801	Cristiano Ronaldo	C. Ronaldo dos Santos Aveiro	https://cdn.sofifa.com/players/020/801/21_60.png	http://sofifa.com/player/20801/c-ronaldo-dos-s...	Portugal	35
2	200389	J. Oblak	Jan Oblak	https://cdn.sofifa.com/players/200/389/21_60.png	http://sofifa.com/player/200389/jan-oblak/210006/	Slovenia	27
3	192985	K. De Bruyne	Kevin De Bruyne	https://cdn.sofifa.com/players/192/985/21_60.png	http://sofifa.com/player/192985/kevin-de-bruyn...	Belgium	29
4	190871	Neymar Jr	Neymar da Silva Santos Jr.	https://cdn.sofifa.com/players/190/871/21_60.png	http://sofifa.com/player/190871/neymar-da-silv...	Brazil	28
...	...	...	...	...	...	...	...
19016	247223	Xia Ao	Ao Xia	https://cdn.sofifa.com/players/247/223/21_60.png	http://sofifa.com/player/247223/ao-xia/210006/	China PR	21
19017	258760	B. Hough	Ben Hough	https://cdn.sofifa.com/players/258/760/21_60.png	http://sofifa.com/player/258760/ben-hough/210006/	England	17
19018	252757	R. McKinley	Ronan McKinley	https://cdn.sofifa.com/players/252/757/21_60.png	http://sofifa.com/player/252757/ronan-mckinley...	England	18
19019	243790	Wang Zhen'ao	Zhen'ao Wang	https://cdn.sofifa.com/players/243/790/21_60.png	http://sofifa.com/player/243790/zhenao-wang/21...	China PR	20
19020	252520	Zhou Xiao	Xiao Zhou	https://cdn.sofifa.com/players/252/520/21_60.png	http://sofifa.com/player/252520/xiao-zhou/210006/	China PR	21

19021 rows × 108 columns

In [ ]:

In [64]:
```
# inspecting my columns
list(dfNew.columns)
```

```
Out[64]: ['ID',
 'Name',
 'LongName',
 'photoUrl',
 'playerUrl',
 'Nationality',
 'Age',
 '↓OVA',
 'POT',
 'Club',
 'Contract',
 'Positions',
 'Height',
 'Weight',
 'Preferred Foot',
 'BOV',
 'Best Position',
 'Joined',
 'Loan Date End',
 'Value',
 'Wage',
 'Release Clause',
 'Attacking',
 'Crossing',
 'Finishing',
 'Heading Accuracy',
 'Short Passing',
 'Volleys',
 'Skill',
 'Dribbling',
 'Curve',
 'FK Accuracy',
 'Long Passing',
 'Ball Control',
 'Movement',
 'Acceleration',
 'Sprint Speed',
 'Agility',
 'Reactions',
 'Balance',
 'Power',
 'Shot Power',
 'Jumping',
 'Stamina',
 'Strength',
 'Long Shots',
 'Mentality',
 'Aggression',
 'Interceptions',
 'Positioning',
 'Vision',
 'Penalties',
 'Composure',
 'Defending',
 'Marking',
 'Standing Tackle',
 'Sliding Tackle',
 'Goalkeeping',
 'GK Diving',
 'GK Handling',
 'GK Kicking',
 'GK Positioning',
 'GK Reflexes',
 'Total Stats',
 'Base Stats',
 'W/F',
 'SM',
 'A/W',
 'D/W',
 'IR',
 'PAC',
 'SHO',
 'PAS',
 'DRI',
 'DEF',
 'PHY',
 'Hits',
 'playerName',
 'playerStatus',
 'CDM',
 'LM',
 'ST',
 'CAM',
 'CM',
 'LB',
 'RWB',
 'RB',
 'LWB',
 'CF',
 'LW',
 'RW',
 'GK',
 'RM',
 'CB',
 'weight',
 'height',
 'W/F1',
 'SM1',
 'IR1',
 'wage',
 'value',
 'Release clause',
 'HITS',
 'height_bins',
 'weight_bins',
 'wage_bins',
 'value_bins',
 'Release clause_bins']
```

In [ ]:

In [65]:

```
# making a copy of my dataframe
df_unClean = dfNew.copy()
```

In [ ]:

In [66]:

```
# removing columns that are not neccessary

# Dropping columns due to data redundancy,and also due to low predicting power
df_unClean = df_unClean.drop(['ID','Name','LongName','photoUrl','playerUrl','Club','Nationality','Contract','Positions'
                 'Joined','Loan Date End','Value','Wage','Release Clause','W/F','IR','Hits','SM', ],axis = 1)
```

In [ ]:

In [67]:

```
df_unClean.columns
```

Out[67]:
```
Index(['Age', '↓OVA', 'POT', 'Preferred Foot', 'BOV', 'Best Position',
       'Attacking', 'Crossing', 'Finishing', 'Heading Accuracy',
       'Short Passing', 'Volleys', 'Skill', 'Dribbling', 'Curve',
       'FK Accuracy', 'Long Passing', 'Ball Control', 'Movement',
       'Acceleration', 'Sprint Speed', 'Agility', 'Reactions', 'Balance',
       'Power', 'Shot Power', 'Jumping', 'Stamina', 'Strength', 'Long Shots',
       'Mentality', 'Aggression', 'Interceptions', 'Positioning', 'Vision',
       'Penalties', 'Composure', 'Defending', 'Marking', 'Standing Tackle',
       'Sliding Tackle', 'Goalkeeping', 'GK Diving', 'GK Handling',
       'GK Kicking', 'GK Positioning', 'GK Reflexes', 'Total Stats',
       'Base Stats', 'A/W', 'D/W', 'PAC', 'SHO', 'PAS', 'DRI', 'DEF', 'PHY',
       'playerName', 'playerStatus', 'CDM', 'LM', 'ST', 'CAM', 'CM', 'LB',
       'RWB', 'RB', 'LWB', 'CF', 'LW', 'RW', 'GK', 'RM', 'CB', 'weight',
       'height', 'W/F1', 'SM1', 'IR1', 'wage', 'value', 'Release clause',
       'HITS', 'height_bins', 'weight_bins', 'wage_bins', 'value_bins',
       'Release clause_bins'],
      dtype='object')
```

In [ ]:

In [68]:
```
# Reassigning the target column on the first index
df_unClean = df_unClean[['HITS','Age', '↓OVA', 'POT', 'Preferred Foot', 'BOV', 'Best Position',
       'Attacking', 'Crossing', 'Finishing', 'Heading Accuracy',
       'Short Passing', 'Volleys', 'Skill', 'Dribbling', 'Curve',
       'FK Accuracy', 'Long Passing', 'Ball Control', 'Movement',
       'Acceleration', 'Sprint Speed', 'Agility', 'Reactions', 'Balance',
       'Power', 'Shot Power', 'Jumping', 'Stamina', 'Strength', 'Long Shots',
       'Mentality', 'Aggression', 'Interceptions', 'Positioning', 'Vision',
       'Penalties', 'Composure', 'Defending', 'Marking', 'Standing Tackle',
       'Sliding Tackle', 'Goalkeeping', 'GK Diving', 'GK Handling',
       'GK Kicking', 'GK Positioning', 'GK Reflexes', 'Total Stats',
       'Base Stats', 'A/W', 'D/W', 'PAC', 'SHO', 'PAS', 'DRI', 'DEF', 'PHY',
       'playerName', 'playerStatus', 'LB', 'RW', 'CAM', 'CM', 'CF', 'CB', 'RM',
       'GK', 'RWB', 'LWB', 'CDM', 'RB', 'ST', 'LM', 'LW', 'W/F1', 'SM1', 'IR1',
       'weight', 'height', 'value', 'wage', 'Release clause',
       'height_bins', 'weight_bins', 'wage_bins', 'value_bins',
       'Release clause_bins']]
```

In [ ]:

In [69]:
```
df_unClean
```

Out[69]:

	HITS	Age	↓OVA	POT	Preferred Foot	BOV	Best Position	Attacking	Crossing	Finishing	...	weight	height	value	wage	Rel cl
0	771.0	33	93.0	93.0	Left	93	RW	429.0	85.0	95.0	...	72.0	170.0	103500000.0	560000.0	1384000
1	562.0	35	92.0	92.0	Right	92	ST	437.0	84.0	95.0	...	83.0	187.0	63000000.0	220000.0	759000
2	150.0	27	91.0	93.0	Right	91	GK	95.0	13.0	11.0	...	87.0	188.0	120000000.0	125000.0	1594000
3	207.0	29	91.0	91.0	Right	91	CAM	407.0	94.0	82.0	...	70.0	181.0	129000000.0	370000.0	1610000
4	595.0	28	91.0	91.0	Right	91	LW	408.0	85.0	87.0	...	68.0	175.0	132000000.0	270000.0	1665000
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
19016	NaN	21	47.0	55.0	Right	49	CB	145.0	23.0	26.0	...	66.0	178.0	100000.0	1000.0	700
19017	NaN	17	47.0	67.0	Right	51	CAM	211.0	38.0	42.0	...	65.0	175.0	130000.0	500000.0	1650
19018	NaN	18	47.0	65.0	Right	49	CAM	200.0	30.0	34.0	...	74.0	179.0	120000.0	500000.0	1310
19019	NaN	20	47.0	57.0	Right	48	ST	215.0	45.0	52.0	...	69.0	175.0	100000.0	2000.0	880
19020	NaN	21	47.0	57.0	Left	50	LB	163.0	40.0	18.0	...	75.0	188.0	100000.0	1000.0	790

19021 rows × 88 columns

In [ ]:

In [70]:

```
# visually inspecting the feature label in the dataset.

cols = df_unClean.columns
for col in cols:
    print('column Name:', col, 'unique values:',df_unClean[col].unique())
```

```
column Name: HITS unique values: [7.71e+02 5.62e+02 1.50e+02 2.07e+02 5.95e+02 2.48e+02 2.46e+02 1.20e+02
 1.60e+03 1.30e+02 3.21e+02 1.89e+02 1.75e+02 9.60e+01 1.18e+02 2.16e+02
 2.12e+02 1.54e+02 2.05e+02 2.02e+02 3.39e+02 4.08e+02 1.03e+02 3.32e+02
 8.60e+01 1.73e+02 1.61e+02 3.96e+02 1.10e+03 4.33e+02 2.42e+02 2.06e+02
 1.77e+02 1.50e+03 1.98e+02 4.59e+02 1.17e+02 1.19e+02 2.09e+02 8.40e+01
 1.87e+02 1.65e+02 2.03e+02 6.50e+01 3.36e+02 1.26e+02 3.13e+02 1.24e+02
 1.45e+02 5.38e+02 1.82e+02 1.01e+02 4.50e+01 3.77e+02 9.90e+01 1.94e+02
 4.03e+02 4.14e+02 5.93e+02 3.74e+02 2.45e+02 3.20e+03 2.66e+02 2.99e+02
 3.09e+02 2.15e+02 2.65e+02 2.11e+02 1.12e+02 3.37e+02 7.00e+01 1.59e+02
 6.88e+02 1.16e+02 6.30e+01 1.44e+02 1.23e+02 7.10e+01 2.24e+02 1.13e+02
 1.68e+02 6.10e+01 8.90e+01 1.37e+02 2.78e+02 7.50e+01 1.48e+02 1.76e+02
 1.97e+02 2.64e+02 2.14e+02 2.47e+02 4.02e+02 4.40e+02 1.70e+03 2.30e+03
 1.71e+02 3.20e+02 6.57e+02 8.70e+01 2.59e+02 2.00e+02 2.55e+02 2.53e+02
 1.96e+02 6.00e+01 9.70e+01 8.50e+01 1.69e+02 2.56e+02 1.32e+02 2.39e+02
 1.66e+02 1.21e+02 1.09e+02 3.20e+01 4.60e+01 1.22e+02 4.80e+01 5.27e+02
 1.99e+02 2.82e+02 5.10e+01 1.90e+03 6.42e+02 1.55e+02 3.23e+02 2.88e+02
 4.97e+02 5.09e+02 7.90e+01 4.90e+01 2.70e+02 5.11e+02 8.00e+01 1.28e+02
 1.15e+02 1.56e+02 2.04e+02 1.43e+02 1.40e+02 1.52e+02 2.20e+02 1.34e+02
 2.25e+02 9.40e+01 7.40e+01 1.35e+02 1.42e+02 5.00e+01 7.70e+01 4.00e+01
 1.07e+02 1.93e+02 1.79e+02 3.40e+01 6.40e+01 4.53e+02 5.70e+01 8.10e+01
 2.80e+01 7.80e+01 1.33e+02 4.30e+01 4.25e+02 8.80e+01 4.20e+01 3.60e+01
 2.33e+02 3.76e+02 2.10e+02 4.44e+02 1.00e+02 2.63e+02 9.80e+01 2.90e+01
 1.60e+02 3.90e+01 2.57e+02 6.00e+00 3.10e+02 1.38e+02 6.20e+01 2.93e+02
 2.85e+02 3.62e+02 6.60e+01 6.90e+01 5.80e+01 2.10e+01 2.00e+01 1.31e+02
 3.80e+01 4.06e+02 6.80e+01 1.08e+02 1.10e+02 9.30e+01 5.12e+02 4.43e+02
 3.06e+02 3.52e+02 4.22e+02 5.85e+02 3.46e+02 1.78e+02 8.41e+02 7.60e+01
 3.94e+02 7.20e+01 1.72e+02 4.40e+01 4.07e+02 2.30e+02 3.67e+02 2.95e+02
 1.57e+02 2.43e+02 5.60e+01 1.11e+02 3.26e+02 6.79e+02 1.80e+01 9.20e+01
 5.90e+01 2.50e+01 1.84e+02 5.30e+01 1.20e+01 9.00e+01 5.50e+01 7.30e+01
 1.10e+01 5.66e+02 1.80e+02 8.30e+01 2.62e+02 1.70e+01 2.60e+01 3.10e+01
 2.80e+02 3.59e+02 2.13e+02 2.97e+02 3.87e+02 4.80e+02 3.81e+02 6.77e+02
 4.86e+02 8.00e+00 2.44e+02 1.29e+02 3.88e+02 2.75e+02 3.19e+02 2.00e+03
 5.20e+01 9.10e+01 4.21e+02 1.53e+02 2.70e+01 4.10e+01 2.22e+02 3.50e+01
 1.02e+02 2.30e+01 3.00e+01 3.30e+01 1.46e+02 1.30e+01 1.90e+01 1.40e+01
 1.06e+02 2.76e+02 5.68e+02 3.53e+02 4.70e+01 4.78e+02 2.49e+02 2.54e+02
 3.69e+02 2.19e+02 5.65e+02 2.37e+02 2.27e+02 4.34e+02 3.75e+02 1.62e+02
 6.05e+02 6.54e+02 3.00e+00 7.00e+00 9.00e+00 1.04e+02 1.14e+02 1.86e+02
 4.46e+02 7.56e+02 2.20e+01 1.39e+02 5.00e+02 6.70e+01 1.47e+02 1.49e+02
 1.60e+01 8.20e+01 5.40e+01 3.70e+01 1.50e+01 1.30e+03 3.00e+03 9.52e+02
 5.00e+00 7.49e+02 5.41e+02 3.30e+02 3.93e+02 5.17e+02 7.70e+02 4.09e+02
 1.70e+02 1.25e+02 2.83e+02 3.42e+02 3.63e+02 5.80e+02 1.05e+02 2.17e+02
 2.40e+01 1.41e+02 1.00e+01 4.27e+02 1.58e+02 4.26e+02 4.00e+00 6.66e+02
 1.81e+02 3.24e+02 9.79e+02 1.40e+03 3.02e+02 7.51e+02 2.98e+02 4.11e+02
 9.44e+02 2.00e+00 9.47e+02 2.92e+02 3.49e+02 6.21e+02 1.00e+00 2.80e+03
 3.38e+02 2.87e+02 2.61e+02 2.18e+02 1.80e+03 2.40e+02 2.79e+02 2.29e+02
 1.88e+02 3.15e+02 6.64e+02 6.13e+02 1.90e+02 7.06e+02 1.27e+02 4.62e+02
 3.86e+02 6.95e+02 4.91e+02 1.67e+02 2.81e+02 2.50e+02 3.07e+02 9.50e+01
 2.31e+02 1.74e+02 6.80e+02 6.33e+02 2.21e+02 3.48e+02 6.02e+02 1.83e+02
 6.53e+02 1.95e+02 1.64e+02 1.51e+02 2.58e+02 8.40e+03 3.43e+02 4.19e+02
 6.55e+02 1.36e+02 3.99e+02 5.31e+02 3.57e+02 2.28e+02 3.85e+02 3.12e+02
 3.40e+02 2.38e+02 4.87e+02 3.55e+02 4.99e+02 4.30e+03 2.96e+02 5.15e+02
 9.43e+02 1.20e+03 9.03e+02 3.35e+02 1.91e+02 5.94e+02 2.67e+02 6.17e+02
 5.16e+02 5.04e+02 3.31e+02 6.52e+02 4.10e+02 5.50e+02 4.73e+02 4.42e+02
 3.44e+02 2.08e+02 1.00e+03 2.50e+03 2.73e+02 4.85e+02 8.26e+02 1.92e+02
 4.05e+02 9.41e+02 4.77e+02 6.44e+02 3.03e+02 4.17e+02 6.00e+03      nan]
column Name: Age unique values: [33 35 27 29 28 31 21 34 32 25 26 30 20 24 22 23 19 38 42 36 37 18 17 39
 40 41 16 43 53]
column Name: ↓OVA unique values: [93. 92. 91. 90. 89. 88. 87. 86. 85. 84. 83. 82. nan 81. 80. 79. 78. 77.
 76. 75. 74. 73. 72. 71. 70. 69. 68. 67. 66. 65. 64. 63. 62. 61. 60. 59.
 58. 57. 56. 55. 54. 53. 52. 51. 50. 49. 48. 47.]
column Name: POT unique values: [93. 92. 91. 90. 95. 89. 88. 87. 86. 85. 84. 83. 82. nan 81. 80. 79. 78.
 77. 76. 75. 74. 73. 72. 71. 70. 69. 68. 67. 66. 65. 64. 63. 62. 61. 60.
 59. 58. 57. 56. 55. 54. 53. 52. 51. 50. 49. 48. 47.]
column Name: Preferred Foot unique values: ['Left' 'Right']
column Name: BOV unique values: [93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70
 69 68 67 66 65 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48]
column Name: Best Position unique values: ['RW' 'ST' 'GK' 'CAM' 'LW' 'CB' 'CDM' 'CF' 'CM' 'RB' 'LB' 'LM' 'RM' 'LWB'
 'RWB']
column Name: Attacking unique values: [429. 437.  95. 407. 408. 423. 392. 114. 118. 316. 410. 349.  86. 119.
 426. 374. 411. 360. 328. 383. 405. 123. 420. 224. 388. 397. 425. 373.
 365. 371. 311. 396. 345. 399. 400.  78. 280. 330. 403. 379. 380.  94.
 394. 419. 339. 293. 344. 390.  84. 359. 372. 377. 346. 389. 386. 308.
 277. 382. 368. 402. 292. 298. 366. 352. 363. 322. 361.  91. 364. 341.
 385. 355. 305. 321. 262.  93. 375. 387. 356. 253. 285. 391. 353. 367.
  90. 295. 378. 256. 338. 331.  69. 105.  85. 358. 343. 319. 271. 113.
 350. 406. 340. 393. 247. 334. 351. 342. 302. 329. 354.  98. 301. 115.
 384. 208.  72. 376.  92. 258. 362.  74. 417.  99. 263.  88. 279. 101.
 395. 100.  81.  87.  55. 310.  82. 117. 409. 318. 323. 248. 315. 381.
 348. 327. 309. 130. 283. 336. 369. 106. 252. 320. 290. 370. 126. 251.
 108. 335. 297. 284.  80.  75. 357. 270.  97. 306. 337.  73. 286. 325.
 326. 324. 333. 103. 259. 273. 313. 296.  61. 312. 347. 401. 304. 278.
  83.  43. 314. 291. 264. 272. 317. 231. 250. 268.  54. 261. 255.  70.
 281. 265. 299. 287.  68. 294.  77. 219. 300. 269. 332. 289. 288. 107.
 282. 122. 244.  89. 112. 274. 276.  nan 307. 229.  96. 109.  76. 125.
 102. 239. 227. 241. 257. 254. 228. 233. 124. 215. 246. 110. 245. 214.
 242. 266. 104.  66. 303. 260.  63. 230. 275.  50. 238. 249. 111.  67.
 240. 221. 237.  56. 235. 234. 243. 267. 232. 203. 223.  64. 213. 222.
 226. 225. 211. 207.  52. 173.  57. 217. 236.  71. 204. 216. 199.  59.
 189.  60. 194. 116. 205. 201. 193.  65. 192. 209. 218. 128. 210.  79.
  45. 206. 162. 220.  49. 197. 202. 212.  58. 190. 181.  51.  62. 200.
 198. 195. 191. 131. 185.  42. 180. 182. 196. 188. 169. 187. 178.  53.
 183. 184. 186. 165. 172.  47. 171. 176. 159.  46. 179. 175. 167. 174.
 161. 170. 177. 164. 134. 168. 163. 166. 158. 150. 143.  48. 152. 160.
 148. 151. 157. 154. 141. 146. 147. 149. 156. 153. 138. 145. 142. 139.
 155. 144. 136. 137.]
column Name: Crossing unique values: [85. 84. 13. 94. 71. 79. 17. 78. 18. 53. 76. 58. 14. 15. 75. 66. 70. 68.
 91. 82. 20. 12. 30. 77. 88. 83. 93. 90. 87. 81. 73. 11. 54. 62. 86. 80.
 55. 42. 57. 65. 63. 64. 52. 40. 69. 47. 60.  9. 16. 44. 72. 50. 56. 46.
 89. 34. 45. 74. 49. 67. 24. 35. 36. 61. 19. 27. 25. 10. 51. 38. 43. 59.
 39. 48. 23.  8. 28. nan 92. 41. 29. 32. 22. 26. 37. 33. 31. 21.  7.  6.]
column Name: Finishing unique values: [95. 11. 82. 87. 94. 91. 13. 14. 52. 90. 64. 88. 65. 85. 66. 84. 10. 22.
 76. 81. 56. 79. 57. 45. 77. 63. 86. 80. 15. 33. 67. 12. 72. 92. 93. 51.
 46. 60. 75. 55. 73. 83. 50. 42. 39. 40.  9. 68. 48. 37. 70. 78. 69.  8.
 53. 89. 25. 62. 71. 74. 44. 26. 19. 32. 18. 61. 58. 30. 54. 36. 29. 16.
 38. 59. 27. 34. 47. 20. 31. 49. 43. 41. 28. nan  5.  7.  6. 21. 17. 35.
 23. 24.  4.  3.]
column Name: Heading Accuracy unique values: [70. 90. 15. 55. 62. 85. 59. 19. 73. 11. 87. 84. 80. 13. 25. 91. 92. 78.
 46. 54. 72. 64. 14. 10. 61. 58. 83. 38. 69. 51. 67. 86. 75. 68. 16. 81.
 21. 79. 53. 65. 82. 12. 42. 48. 88. 66. 76. 74. 52. 23. 40. 49. 60. 44.
 20. 37. 71. 17. 45. 77. 50. 63. 43. 39. 57. 56. 47. 24. 18. 31. 28. 35.
 34. 41. 36. 93.  7. nan 30. 89.  8. 26. 33. 27. 32. 22. 29.  9.  5.  6.]
column Name: Short Passing unique values: ['91' '82' '43' '94' '87' '84' '45' '83' '61' '79' '85' '33' '55' '86'
 '57' '81' '42' '74' '93' '88' '30' '65' '89' '77' '32' '50' '80' '78'
 '90' '69' '40' '92' '75' '73' '34' '76' '35' '70' '37' '23' '44' '38'
 '48' '26' '60' '25' '46' '28' '24' '36' '51' '17' '18' '39' '71' '67'
 '27' '72' '66' '20' '31' '68' '29' '11' '64' '62' nan '41' '63' '19' '54'
 '16' '69_' '22' '49' '59' '14' '58' '15' '21' '52' '56' '53' '12' '47'
 '13' 58 65 70 67 66 57 72 37 26 60 64 55 56 59 68 21 74 42 63 62 47 22 52
 15 12 73 61 54 71 25 28 27 31 69 50 75 29 36 41 32 53 48 30 11 35 16 51
 18 43 19 34 33 23 38 20 13 39 49 24 46 17 8 14 45 44 40 76 7]
column Name: Volleys unique values: [88. 86. 13. 82. 87. 89. 79. 20. 83. 14. 45. 75. 63. 12. 11. 69. 67. 56.
 18. 85. 62. 70. 32. 40. 47. 81. 44. 84. 78. 76. 90. 49. 42. 64. 57. 60.
  8. 72. 71. 59. 74. 80. 73. 37. 31. 38. 61. 10. 77. 68. 58. 66. 30. 33.
 65. 27. 51. 15. 16. 50. 43. 35. 24. 17. 34. 28.  9. 39. 52. 46. 22. 19.
 53. 55. 48. 54. 23.  5. 41. 25. 21. 36. nan 26. 29.  6.  7.  4.  3.]
column Name: Skill unique values: [470. 414. 109. 441. 448. 407. 406. 138. 394. 144. 363. 391. 369. 110.
 160. 404. 381. 397. 387. 336. 400. 436. 157. 395. 100. 262. 427. 432.
 429. 380. 426. 411. 358. 351. 433. 365. 403.  98. 276. 386. 383.  99.
 413. 115. 341. 375. 143. 359. 309. 435. 330. 325. 355.  96. 420. 412.
 388. 319. 269. 399. 106. 402. 425. 297. 312. 418. 372. 352. 439. 409.
 349. 116. 371. 428. 104. 345. 430. 295. 405. 440. 422. 252. 401. 417.
 396. 233. 377. 251. 382. 368.  84. 356. 342. 410. 271. 350.  83. 126.
 103. 370. 362. 343. 328. 344. 415. 378. 275. 416. 119. 127. 373. 384.
  77. 393. 348. 317. 408. 376. 300. 220.  89. 107. 334.  72. 390. 419.
 305. 289. 398. 281. 354. 102. 339. 385. 139. 292.  97. 421.  91. 105.
```

```
  73. 335. 101. 340. 337. 306. 113. 122. 123. 302. 364. 250. 347. 333.
 323. 389. 361. 322.  86. 367. 258. 392.  92.  90. 310. 331. 338. 121.
 260.  82. 245. 324. 346. 379. 299. 284. 320. 283. 108. 278. 286. 296.
 315. 274.  88. 114. 264. 288.  94. 326. 366. 117. 360. 424.  93. 318.
 124. 125. 327. 249.  75. 332. 303. 374. 239. 272. 357. 353. 266. 321.
 277. 268. 314. 294. 240.  95. 227. 112. 118. 263. 280. 140. 282.  81.
 329. 201.  87. 221. 257. 285. 316. 287. 307. 270. 256. 313. 311.  nan
 228. 247. 254. 130.  80.  85. 232. 293. 298. 301. 213. 168. 291. 216.
 290. 308. 261. 171. 267. 242. 219. 248. 237. 243. 279. 246. 273.  78.
 255. 253. 230.  74. 210. 235. 231. 208. 259. 304. 241. 199. 224. 206.
  61. 129. 222. 223. 141. 149. 131. 225.  71. 189. 265. 226.  70. 179.
 192. 134. 209. 173. 234.  76. 236. 212.  69. 218. 120. 177. 238. 204.
 229. 215. 165. 211. 195.  64. 202. 194. 190. 193. 203.  67. 214.  79.
 205. 244. 196. 111. 187.  65. 200.  63. 198. 217. 135.  68. 184. 167.
 148. 207. 142. 185. 133. 191. 181. 197.  43.  66. 175. 182.  51. 180.
 169. 186. 137. 188. 176. 132.  60. 178. 147. 163. 183. 162. 152. 170.
 172. 174. 159. 161. 154. 153. 128.  62. 166.  53. 155.  56. 151. 164.
 158.  46. 150.  59.  55.  58. 156. 146.  52. 136.  54.  47.  48. 145.
  40.  57.]
column Name: Dribbling unique values: ['96' '88' '12' '95' '85' '90' '27' '92' '21' '70' '91' '69' '13' '30'
 '87' '65' '79' '83' '23' '80' '18' '93' '77' '63' '76' '16' '59' '81'
 '11' '84' '10' '75' '78' '55' '15' '86' '66' '67' '28' '57' '64' '82'
 '62' '19' '53' '72' '50' '26' '43' '89' '73' '20' '14' '68' '71' '74'
 '22' '54' '56' '61' '9' '24' '60' '25' '8' '17' '47' '58' '46' '42' '51'
 '52' '49' '44' '35' '48' '39' '29' '40' '70_' '45' nan '34' '31' '33'
 '38' '41' '32' '7' '37' '36' '5' '6' 64 46 65 61 57 60 71 19 41 18 63 70
 74 62 58 12 67 52 26 50 13 32 33 14 75 49 51 76 16 36 59 34 68 66 72 17
 54 44 73 42 56 55 37 69 40 30 47 24 11 15 45 35 39 38 6 7 53 43 48 10 20
 29 9 8 28 5 31 25 22 27 23 77 21]
column Name: Curve unique values: [93. 81. 13. 85. 88. 79. 83. 19. 18. 60. 76. 63. 14. 74. 77. 49. 15. 80.
 12. 28. 86. 84. 82. 61. 71. 11. 66. 16. 89. 70. 21. 46. 78. 67. 58. 65.
 48. 34. 90. 59. 55. 87. 62.  9. 56. 36. 30. 32. 73. 69. 68. 75. 45. 10.
 72. 64. 41. 23. 47. 20. 51. 25. 44. 17. 54. 57. 53. 33. 40. 50. 39. 35.
 52. 42. 37. 43. 26. 31. 92. 91. nan 29. 94. 27. 38. 22. 24.  8.  6.  7.
  5.  4.]
column Name: FK Accuracy unique values: [94. 76. 14. 83. 89. 85. 69. 18. 63. 12. 70. 64. 74. 20. 11. 73. 49. 61.
 88. 68. 28. 79. 84. 48. 67. 38. 87. 53. 65. 15. 31. 78. 82. 10. 51. 59.
 19. 47. 52. 57. 43. 13. 77. 54. 75. 86. 55. 30. 62. 32. 58. 93.  8. 66.
 71. 81. 92. 44. 17. 60. 40. 16. 72. 46. 35. 45. 29. 21. 56. 80. 24. 22.
 39. 42. 26. 41.  9. 37. 27. 50. 33. 25. 36. 91. 34. 23. nan  7.  6. 90.
  5.]
column Name: Long Passing unique values: [91. 77. 40. 93. 81. 70. 75. 44. 63. 86. 71. 84. 35. 59. 73. 83. 64. 69.
 79. 82. 68. 89. 76. 80. 87. 37. 65. 36. 50. 53. 78. 47. 74. 48. 31. 85.
 24. 55. 90. 54. 62. 32. 49. 66. 67. 51. 28. 46. 52. 72. 56. 41. 45. 22.
 88. 61. 33. 12. 60. 17. 27. 29. 23. 38. 16. 58. 34. 25. 39. 21. 30. 42.
 43. 57. 20. 26. nan 18. 19. 13. 15. 11. 14.  9. 10.  5.  8.]
column Name: Ball Control unique values: [96. 92. 30. 95. 88. 89. 90. 77. 79. 23. 46. 83. 80. 85. 94. 40. 84. 16.
 74. 91. 87. 82. 78. 19. 61. 22. 34. 38. 81. 25. 86. 76. 69. 28. 93. 75.
 35. 60. 63. 73. 18. 71. 15. 21. 72. 14. 65. 20. 24. 27. 70. 33. 17. 62.
 64.  9. 68. 67. 32. 26. 66. 52. 11. 57. 58. 29. 12. 37. 10. 36. 13. 31.
 55. 59. 39. nan 54. 56. 48. 44. 51. 50. 47. 49. 53.  5. 42.  8. 45. 43.
 41.  7.]
column Name: Movement unique values: [451. 431. 307. 398. 453. 407. 460. 268. 458. 254. 354. 343. 284. 286.
 388. 378. 424. 464. 420. 399. 437. 322. 367. 272. 328. 448. 332. 425.
 435. 391. 434. 400. 331. 349. 429. 416. 312. 326. 418. 419. 417. 386.
 321. 409. 374. 304. 403. 351. 401. 365. 414. 292. 323. 299. 433. 350.
 348. 413. 320. 281. 427. 353. 364. 410. 428. 316. 381. 442. 375. 288.
 395. 385. 251. 319. 444. 383. 298. 411. 412. 415. 393. 397. 443. 423.
 387. 422. 327. 390. 362. 352. 406. 277. 361. 421. 396. 384. 450. 338.
 363. 359. 287. 297. 430. 382. 377. 380. 438. 449. 257. 371. 339. 341.
 404. 345. 394. 295. 246. 265. 258. 366. 294. 314. 266. 405. 218. 337.
 267. 220. 376. 309. 283. 426. 347. 244. 240. 291. 340. 250. 305. 290.
 317. 334. 355. 333. 389. 330. 318. 441. 402. 344. 335. 219. 264. 408.
 274. 373. 379. 256. 229. 392. 372. 360. 262. 346. 278. 248. 368. 279.
 269. 336. 342. 236. 370. 243. 315. 249. 227. 329. 239. 369. 223. 282.
 358. 271. 313. 270. 356. 263. 184. 311. 436. 432. 221. 301. 190. 259.
 308. 235. 260. 217. 275. 285. 210. 234. 276. 310. 447. 180. 446. 300.
 303. 209. 247. 252. 231. 357. 226. 238. 280. 440. 237. 245. 296. 325.
 273. 306. 196. 242. 199. 178. 222. 445.  nan 324. 293. 302. 289. 214.
 192. 206. 225. 197. 241. 230. 188. 202. 208. 203. 216. 213. 224. 439.
 212. 232. 253. 228. 189. 204. 205. 207. 198. 168. 255. 215. 194. 191.
 185. 145. 261. 156. 201. 193. 181. 233. 195. 183. 152. 211. 160. 173.
 170. 176. 147. 143. 159. 187. 169. 200. 165. 163. 177. 179. 167. 139.
 162. 175. 155. 166. 172. 174. 154. 164. 182. 150. 186. 146. 138. 157.
 137. 135. 171. 158. 161. 149. 124. 144. 151. 148. 141. 134. 153. 126.
 142. 125. 132. 127. 140. 133. 130. 131. 136. 122.]
column Name: Acceleration unique values: [91. 87. 43. 77. 94. 56. 96. 38. 72. 95. 60. 42. 54. 79. 89. 64. 66. 51.
 73. 57. 80. 86. 85. 78. 40. 82. 76. 65. 68. 90. 48. 46. 88. 70. 83. 84.
 93. 52. 74. 92. 55. 58. 59. 67. 81. 62. 44. 71. 69. 50. 53. 45. 49. 75.
 41. 61. 63. 35. 47. 34. 36. 37. 39. 30. 97. 31. 33. 32. 27. 28. nan 26.
 29. 25. 17. 19. 24. 15. 23. 21. 20. 22. 16. 18. 13. 14.]
column Name: Sprint Speed unique values: [80. 91. 60. 76. 89. 78. 92. 47. 96. 50. 79. 93. 69. 52. 72. 70. 90. 66.
 82. 63. 55. 77. 86. 81. 83. 85. 65. 68. 53. 43. 94. 62. 58. 61. 87. 64.
 67. 54. 88. 75. 95. 73. 49. 84. 56. 44. 74. 51. 57. 46. 59. 71. 37. 34.
 33. 42. 30. 35. 48. 39. 45. 40. 18. 38. 41. 27. 32. 29. nan 28. 36. 26.
 31. 22. 25. 23. 15. 20. 17. 16. 24. 19. 21. 12. 14.]
column Name: Agility unique values: [91. 87. 67. 78. 96. 77. 40. 92. 37. 61. 93. 51. 79. 84. 94. 82. 60. 69.
 47. 52. 63. 74. 59. 66. 86. 85. 57. 55. 76. 75. 73. 62. 72. 90. 68. 64.
 80. 56. 48. 83. 41. 81. 54. 88. 33. 65. 49. 71. 89. 45. 70. 43. 50. 32.
 42. 39. 58. 36. 34. 53. 46. 95. 44. 38. 21. 29. 35. 31. 19. nan 26. 30.
 22. 28. 24. 25. 23. 27. 14. 18. 15. 20.]
column Name: Reactions unique values: [94. 95. 88. 91. 93. 92. 86. 89. 87. 84. 90. 83. 85. 82. 81. 79. 80. 74.
 75. 78. 77. 73. 76. 71. 70. 68. 72. 66. 69. 65. 67. 64. 59. nan 60. 62.
 63. 61. 58. 57. 56. 50. 54. 53. 55. 52. 32. 49. 48. 45. 51. 46. 47. 37.
 34. 44. 40. 38. 43. 41. 35. 42. 33. 39. 31. 36. 30. 24. 29. 28.]
column Name: Balance unique values: [95. 71. 49. 76. 83. 82. 91. 37. 43. 53. 86. 66. 45. 35. 69. 94. 92. 84.
 90. 48. 73. 36. 41. 93. 74. 60. 79. 65. 78. 61. 57. 50. 68. 51. 54. 77.
 81. 39. 75. 58. 87. 85. 63. 38. 88. 67. 72. 62. 80. 44. 46. 42. 55. 40.
 70. 32. 89. 52. 59. 47. 64. 27. 56. 30. 31. 25. 34. 29. 24. 96. 33. 28.
 20. nan 23. 22. 26. 21. 17. 97. 19. 12. 18.]
column Name: Power unique values: [389. 444. 268. 408. 357. 420. 393. 240. 404. 402. 406. 437. 249. 284.
 400. 403. 358. 381. 382. 273. 424. 264. 316. 361. 355. 328. 370. 350.
 365. 348. 411. 395. 385. 257. 337. 250. 379. 371. 409. 223. 398. 388.
 241. 347. 308. 426. 378. 343. 341. 262. 325. 345. 359. 399. 421. 396.
 315. 253. 368. 336. 340. 366. 387. 369. 375. 260. 326. 346. 373. 412.
 364. 279. 376. 372. 415. 356. 333. 338. 342. 410. 407. 430. 394. 354.
 331. 239. 234. 392. 270. 422. 374. 360. 391. 300. 335. 242. 327. 215.
 397. 321. 390. 339. 383. 265. 288. 224. 351. 252. 429. 416. 380. 413.
 377. 405. 349. 232. 386. 362. 192. 320. 251. 329. 271. 237. 427. 259.
 255. 266. 227. 353. 258. 243. 263. 291. 302. 306. 332. 363. 256. 247.
 301. 287. 322. 419. 312. 245. 297. 401. 344. 235. 289. 233. 317. 334.
 216. 367. 352. 318. 226. 324. 219. 319. 292. 244. 423. 323. 304. 208.
 314. 313. 193. 299. 303. 311. 229. 211. 225. 309. 330. 238. 305. 220.
 296. 212. 231. 283. 207. 198. 281. 384. 307. 272. 298. 248. 310. 267.
 214. 282. 274. 280. 230. 228. 221. 277. 276. 285. 290. 269. 246.  nan
 294. 293. 195. 236. 295. 217. 189. 275. 201. 278. 194. 206. 218. 176.
 205. 185. 196. 222. 204. 188. 197. 209. 286. 168. 254. 200. 183. 179.
 159. 180. 187. 164. 178. 190. 213. 202. 186. 191. 261. 210. 203. 173.
 199. 169. 152. 181. 175. 184. 182. 170. 160. 162. 167. 177. 139. 161.
 172. 165. 171. 128. 174. 158. 153. 166. 155. 163. 151. 122. 142. 143.
 156. 149. 144. 157. 147. 154. 150. 134. 140.]
column Name: Shot Power unique values: [86. 94. 59. 91. 80. 89. 64. 66. 81. 84. 88. 56. 68. 79. 78. 71. 82. 70.
 55. 76. 61. 83. 51. 52. 90. 87. 62. 72. 77. 74. 50. 57. 58. 85. 60. 75.
 67. 65. 93. 46. 54. 69. 41. 73. 40. 53. 95. 43. 63. 42. 48. 31. 44. 37.
 49. 39. 45. 38. 47. 30. 33. 25. 34. nan 36. 28. 27. 32. 26. 35. 23. 22.
 29. 20. 24. 21. 18.]
column Name: Jumping unique values: [68. 95. 78. 63. 62. 84. 69. 52. 77. 79. 90. 86. 87. 93. 57. 75. 66. 82.
 56. 32. 51. 76. 72. 81. 74. 71. 67. 65. 73. 64. 70. 80. 85. 37. 89. 60.
 49. 50. 83. 58. 53. 59. 88. 38. 92. 34. 61. 46. 43. 36. 91. 39. 45. 42.
 40. 54. 33. 55. 31. 44. 35. 47. 48. 30. 41. 94. nan 28. 29. 27. 24. 19.
 26. 17. 15. 22.]
column Name: Stamina unique values: [72. 84. 41. 89. 81. 76. 85. 32. 86. 35. 75. 88. 90. 38. 43. 78. 79. 96.
 95. 70. 82. 77. 93. 94. 87. 39. 54. 80. 45. 83. 69. 65. 73. 91. 34. 66.
 71. 92. 62. 67. 64. 63. 68. 36. 61. 74. 42. 40. 23. 44. 31. 57. 20. 37.
 29. 30. 56. 60. 52. 48. 58. 25. 51. 26. 27. 59. 28. 53. 33. 49. 97. 55.
 nan 50. 46. 24. 21. 22. 15. 47. 17. 19. 16. 18. 14. 12.]
column Name: Strength unique values: [69. 78. 74. 50. 86. 75. 76. 92. 70. 91. 80. 85. 65. 72. 67. 60. 84. 71.
 94. 63. 73. 62. 54. 81. 64. 87. 58. 43. 77. 66. 53. 89. 68. 46. 44. 61.
 79. 88. 59. 83. 55. 34. 82. 95. 56. 37. 90. 57. 93. 49. 39. 51. 52. 40.
 48. 41. 47. 35. 42. 33. 45. 32. 38. 30. nan 31. 36. 29. 27. 24. 28. 16.
 97. 96. 20. 25. 26. 23.]
column Name: Long Shots unique values: [94. 93. 12. 91. 84. 85. 14. 79. 10. 64. 78. 81. 17. 16. 65. 87. 18. 86.
 19. 15. 82. 63. 74. 76. 47. 89. 70. 90. 77. 13. 49. 54. 88. 80. 53. 58.
 51. 73. 66. 75. 83. 30. 46. 35. 71. 61. 72. 69. 43. 48. 62. 41. 60. 11.
 26. 57. 59. 68. 67.  7. 27. 56. 20. 52. 92. 50. 22. 40. 39. 44. 31. 42.
  9.  6. 55. 28. 23. 38. 24. 25. 34. 36. 29.  4.  8. 45. 33. 37. 21. nan
 32.  5.]
column Name: Mentality unique values: [347. 353. 140. 408. 356. 391. 376. 341. 171. 358. 396. 122. 188. 363.
 414. 332. 386. 379. 348. 172. 382. 123. 294. 378. 313. 371. 331. 412.
 345. 377. 161. 306. 387. 339. 135. 360. 138. 369. 359. 170. 361. 321.
 397. 394. 385. 366. 162. 337. 362. 344. 319. 315. 144. 336. 340. 373.
 398. 324. 300. 338. 384. 139. 364. 372. 134. 354. 342. 308. 322. 383.
 263. 149. 304. 367. 357. 390. 291. 279. 310. 388. 375. 349. 351. 365.
 133. 334. 303. 380. 153. 392. 169. 318. 350. 352. 401. 302. 325. 346.
 132. 399. 281. 335. 403. 307. 368. 141. 126. 328. 245. 131. 320. 127.
 421. 400. 137. 374. 305.  92. 316. 311. 120. 389. 145. 355. 148. 343.
 142. 130. 121. 157. 329. 323. 115. 150. 298. 154. 317. 295. 100. 301.
 326. 327. 197. 273. 287. 370. 290. 103. 393. 312. 297.  89. 271. 299.
 124. 333. 258. 309. 158. 272. 118. 314. 330. 292. 404. 101. 280. 277.
 296. 248. 285. 278. 109.  93. 146. 286. 284. 288. 105. 152. 111. 160.
 119. 156.  95.  99. 238. 104. 266. 276. 275. 265. 106. 254. 293. 282.
 168. 260. 136. 102. 267. 113. 289.  96. 270. 176. 164. 128. 268. 283.
 244. 182.  nan 243. 240. 116. 264. 112. 274. 261. 114. 269. 110. 257.
 179. 155. 252. 262. 151. 247. 108. 256. 117. 249. 253. 231. 159. 163.
  84. 251.  97.  91.  75. 147. 129. 230. 242. 250. 259. 125. 381.  77.
 175.  82.  88.  90. 165.  83. 195.  87. 246. 255.  85.  94. 226. 216.
 236. 220. 107. 241. 228. 198. 239. 225. 181. 233. 219. 166. 183.  98.
 237. 235.  86. 229. 217. 143. 232. 209. 234. 224. 206. 227. 222.  80.
  78. 186. 221. 173. 214. 187.  79.  68. 167.  81. 218. 212. 199. 210.
  74. 223. 208. 213. 201.  72. 215. 202. 205. 203. 204. 190.  76. 211.
 207. 192.  70. 194. 196. 189.  66. 193. 200.  67. 191. 184.  71.  64.
  65.  69. 177.  63.  73.  51.  58. 180. 185. 174.  60.  55. 178.  62.
  50.  59.]
column Name: Aggression unique values: [44 63 34 76 51 81 27 62 43 83 75 91 23 29 90 65 59 89 48 38 25 87 54 60
 73 74 69 85 70 86 32 40 31 77 84 80 78 79 71 56 42 30 61 58 28 82 46 52
 36 92 55 35 67 37 72 57 50 64 39 47 20 68 15 66 33 93 88 22 24 45 17 18
 26 21 11 41 53 19 12 49 94 16 95 13 14 96 10  9]
column Name: Interceptions unique values: [40. 29. 19. 66. 36. 49. 55. 11. 38. 22. 90. 35. 87. 15. 30. 39. 88. 24.
 91. 82. 42. 27. 41. 79. 74. 58. 20. 85. 48. 83. 64. 21. 50. 81. 78. 28.
 86. 26. 34. 52. 37. 80. 25. 56. 23. 47. 45. 77. 84. 44. 53. 18. 46. 72.
 61. 89. 54. 63. 65. 73. 16. 32. 76. 59. 13. 70. 31. 69. 33. 17. 75. 68.
 60. 51. 71. 12. 57. 10. 43. 67. 14.  9. 62. nan  8.  7.  6.  4.  5.  3.]
column Name: Positioning unique values: [93. 95. 11. 88. 87. 94. 91. 13. 47. 92. 72. 12. 90. 73. 80. 85. 20. 35.
 76. 89. 83. 77. 54. 70. 86. 16. 28. 14. 84. 78. 10. 75. 52. 71. 81. 64.
 56. 15. 82. 79. 44. 30. 59.  7. 68. 38. 48. 67. 24. 26. 34. 69. 74. 32.
 66. 62. 65. 51. 18. 31.  9. 25. 49. 55. 63. 27. 61. 17. 39. 58. 29. 50.
 40. 19.  8. 42. 60. 57. 37. 45. 43. 53.  5.  4. 36.  6. 46. 41. 23. 22.
 33. nan 21.  3.  2.]
column Name: Vision unique values: [95 82 65 94 90 79 84 66 80 70 85 44 87 71 83 41 52 86 68 50 77 48 88 30
 61 74 59 73 72 64 91 78 63 57 89 62 56 69 42 67 27 76 81 55 75 60 49 45
 58 22 53 46 25 43 51 40 93 33 31 34 35 39 47 21 32 28 37 36 38 54 24 23
 14 11 15 26 19 18 12 20 17 10 29 13 16  9]
column Name: Penalties unique values: [75. 84. 11. 92. 88. 83. 23. 70. 25. 62. 71. 66. 27. 47. 69. 54. 44. 86.
 17. 90. 33. 87. 73. 60. 55. 68. 91. 72. 50. 78. 18. 82. 40. 29. 45. 43.
 64. 24. 59. 46. 56. 81. 67. 49. 61. 74. 58. 63. 79. 38. 80. 32. 20. 76.
 77. 41. 19. 26. 85. 21. 52. 34. 53. 65. 57. 16. 42. 89. 15. 13. 14. 22.
 51. 37.  9. 48. 12. 31. 36. 39. 10. 30. 35. nan 28.  8.  7.  6.]
column Name: Composure unique values: [96. 95. 68. 91. 93. 88. 90. 65. 84. 70. 66. 80. 85. 69. 82. 89. 81. 87.
 83. 86. 67. 92. 94. 57. 78. 79. 75. 45. 61. 76. 58. 62. 77. 74. 59. 55.
 48. 40. 64. 73. 39. 71. 72. 63. 60. 52. 53. 56. 44. 54. 41. 32. nan 49.
 46. 31. 51. 50. 25. 18. 38. 30. 24. 21. 36. 33. 26. 23. 47. 22. 28. 34.
 35. 37. 43. 27. 12. 42. 17. 29. 13. 19. 14. 16. 20. 15.]
column Name: Defending unique values: [ 91.  84.  57. 186.  94.  96. 122.  50. 100.  48. 272. 259.  54.  38.
  89. 263.  83. 147. 264. 245. 120.  52. 130. 267. 205. 162. 105. 241.
 148. 248. 266. 194. 258. 117. 166.  56. 249.  92.  45. 214. 140.  99.
 150.  59. 251. 262. 243. 195. 160.  40. 114. 236. 244. 231.  80. 123.
 253. 132. 103. 257. 261.  98.  78. 209. 229. 230.  60. 101. 206. 242.
 138.  61. 256. 171. 260. 226. 224.  44. 131. 113. 240.  77. 232. 225.
 109. 228. 247.  93. 121. 238. 111. 128. 188. 173. 250. 255.  41. 144.
 239. 217. 106. 165. 246. 235. 126. 118. 203. 234. 135. 215. 175. 192.
 108.  39.  33. 151. 156. 174.  47. 216. 237. 102. 227. 161. 233.  67.
 213.  75. 212.  36. 254. 196.  88.  81. 134.  53. 155. 223.  43. 125.
  46.  51. 137.  71.  95.  35. 208. 110. 170.  87. 107.  55. 204. 177.
  69. 152. 163.  37. 181. 252. 159. 133. 124. 207.  82.  97.  65.  42.
  79. 104. 211. 129.  49. 157. 153. 185. 189. 146.  86. 112.  73. 127.
  31. 220. 164. 191. 219. 139.  64. 183.  66. 197.  90.  nan 218.  34.
  72. 221. 222. 142.  63. 136. 179.  85. 169. 180.  74. 210.  62. 187.
 145. 198. 184. 199.  32.  30.  58. 172. 178. 116. 176.  70. 202. 141.
 115. 193. 149.  29. 201. 167. 168. 182. 119. 190. 200.  76. 143. 158.
 154.  68.  28.  27.  25.  24.  26.  23.  20.  21.]
column Name: Marking unique values: [32. 28. 27. 68. 35. 38. 15. 34. 25. 93. 42. 84. 20. 17. 47. 85. 30. 89.
 82. 29. 56. 91. 72. 59. 79. 49. 83. 86. 50. 60. 94. 41. 57. 78. 63. 88.
 90.  9. 58. 74. 39. 92. 45. 36. 44. 87. 70. 76. 53. 80. 67. 77. 12. 48.
 55. 75. 81. 11. 64. 69. 14. 24. 52. 65. 19. 31. 13. 10. 66. 71. 54. 46.
 22. 40. 18. 51. 37. 43. 61. 26. 73. 21.  7. 33. 62. 16. 23. nan  8.  6.
```

```
  5.  4.  3.]
column Name: Standing Tackle unique values: [35. 32. 12. 65. 30. 42. 43. 19. 34. 13. 93. 88. 18. 10. 24. 29. 53. 90.
 84. 48. 15. 36. 89. 27. 73. 54. 41. 83. 59. 67. 87. 64. 14. 55. 75. 45.
 33. 57. 21. 82. 50. 86. 80. 79. 31. 46. 85. 40. 44. 56. 20. 70. 76. 81.
 71. 16. 68. 37. 38. 78. 39. 77. 11. 74. 28. 49. 47. 72. 61. 51. 22. 17.
 52. 63. 23. 60. 25. 26. nan  9. 62. 58. 66. 69.  7.  8.  6.  5.]
column Name: Sliding Tackle unique values: [24. 18. 53. 29. 19. 41. 16. 32. 10. 86. 38. 87. 11. 90. 47. 85. 79. 40.
  8. 13. 22. 60. 49. 81. 88. 55. 33. 42. 14. 80. 36. 12. 52. 71. 46. 83.
 65. 84. 34. 82. 77. 78. 74. 20. 43. 35. 69. 70. 30. 68. 45. 57. 44. 21.
 75. 26. 51. 76. 39. 48. 28. 63. 59. 66. 72. 17. 67. 64. 31. 25. 15. 54.
 58. 62. 56. 23. 37. 73. 50. 27. nan  9. 61.  7.  6.  4.]
column Name: Goalkeeping unique values: [ 54.  58. 437.  56.  59.  51.  62. 439.  42.  67. 420. 440.  41.  46.
  63.  60.  26. 435. 424.  43.  45.  52.  50.  47.  53.  44. 418.  15.
  48. 416. 153. 413.  65.  64.  20. 423.  49.  55.  66.  40.  57. 421.
  13.  39.  61. 419.  21. 409.  37. 406. 410.  36. 408.  34.  29. 405.
 403. 402. 407.  16.  69. 391. 401. 398. 400.  22.  68. 396.  38.  78.
  73. 399. 390. 393. 395. 397.  80.  70. 389.  nan 394. 388.  27.  30.
  75.  71. 386.  74. 378. 385. 384. 380. 392. 381.  10. 387. 383. 375.
 382.  19. 379.  24. 369. 356. 368. 373. 370. 372.  72. 374. 376. 364.
  25. 367.  17. 377. 371. 365. 352. 362. 359. 363. 366.  82.  35. 361.
 358.  76. 294.  83. 357. 360. 355. 354.  77. 229. 350. 353. 347. 351.
  32. 349. 169. 346. 348. 343. 345. 339. 342.  33. 341.  28. 119. 337.
 338. 340. 344. 335.  98. 324. 248. 334. 298. 336. 328. 331. 321. 332.
  81.  79. 333. 278. 329. 261. 325.  31. 327. 330. 322. 305. 326. 283.
 320. 323. 318.  18. 319. 316. 317. 272. 315.  88. 311. 310. 314. 313.
 307. 312. 309. 308. 301. 304. 292. 303. 306. 296. 289. 300. 302. 297.
 290. 299. 293. 295. 291. 288.  93. 284. 287. 286. 285. 273. 282. 279.
 281. 280. 277. 275. 276. 274. 270. 271. 268. 269. 267. 260. 265. 262.
 266. 263. 264. 251. 259. 254. 257. 252. 255. 256. 258. 247. 250. 243.
 253. 249. 245. 236. 246. 234. 241. 231.]
column Name: GK Diving unique values: [ 6.  7. 87. 15.  9. 14. 86. 13. 88. 10. 84. 11.  8.  5. 12. 90.  3. 27.
 89. 80. 16. 85.  2. 82. 79. 83.  4. 81. 77. 18. 78. 17. nan 75. 74. 76.
 73. 71. 72. 52. 68. 70. 54. 69. 32. 66. 65. 67. 61. 22. 64. 23. 40. 63.
 55. 19. 50. 62. 58. 60. 59. 56. 57. 53. 51. 49. 46. 48. 47. 45.]
column Name: GK Handling unique values: [11. 92. 13.  9.  6. 14. 88.  5. 85. 10. 89. 87.  8. 15. 12.  4. 82. 81.
  3.  7. 25. 86. 83.  2. 80. 16. 77. 79. 78. 76. 84. 75. 72. nan 74. 71.
 69. 73. 70. 67. 68. 65. 61. 62. 64. 41. 63. 66. 33. 22. 17. 57. 18. 54.
 55. 59. 49. 19. 40. 60. 58. 43. 45. 53. 47. 56. 51. 52. 50. 48. 46.]
column Name: GK Kicking unique values: [15. 78.  5. 12.  9. 85.  7. 88. 13. 16. 74. 91.  6. 10.  4. 93. 11. 73.
 14. 75.  2. 31. 68. 76.  8. 80. 82.  3. 87. 72. 83. 77. 79. 81. 69. 71.
 20. 67. 70. nan 64. 65. 63. 44. 60. 84. 54. 48. 61. 18. 66. 17. 59. 62.
 90. 43. 38. 58. 57. 28. 40. 53. 23. 47. 46. 19. 51. 55. 52. 56. 22. 30.
 25. 42. 35. 21. 49. 50. 36. 45.]
column Name: GK Positioning unique values: [14. 90. 10. 15.  8. 11. 91. 88.  7. 12. 85. 86.  5. 89. 13.  6. 82.  4.
  9. 87. 33. 84. 16. 83.  2.  3. 79. 81. 80. 76. 78. 19. 77. 17. nan 75.
 74. 73. 71. 18. 72. 70. 69. 66. 68. 40. 64. 20. 32. 67. 62. 65. 63. 24.
 23. 50. 55. 58. 51. 59. 56. 61. 57. 60. 46. 54. 53. 52. 47. 49. 48. 43.
 45. 42. 38. 44. 41.]
column Name: GK Reflexes unique values: [ 8 11 90 13 10 14 89  6 12 88  7  9 15  5  3 37 85 86  4 16 82 83 84 87
 78 80 20 18 79 81 19 77 17  2 74 71 76 73 75 72 69 46 66 51 70 34 67 23
 68 45 65 21 59 54 47 61 64 63 62 60 58 56 57 55 53 50 52 49 48 44]
column Name: Total Stats unique values: [2231. 2221. 1413. ...  757.  747.  956.]
column Name: Base Stats unique values: [466 464 489 485 451 457 470 490 484 455 469 463 468 497 442 439 473 452
 498 449 477 401 446 447 465 430 461 422 476 460 453 467 471 399 424 441
 459 438 437 454 428 445 431 474 421 435 448 475 403 444 443 419 405 420
 423 396 388 482 478 385 394 480 433 450 462 456 436 434 429 400 440 425
 410 458 398 413 373 406 408 472 426 407 432 427 415 481 417 372 380 418
 383 414 409 412 411 386 362 402 390 404 391 416 375 389 361 397 366 392
 393 382 368 387 352 376 384 378 379 341 354 369 395 357 381 377 344 360
 370 338 333 367 363 349 355 345 358 348 374 351 343 342 353 321 350 365
 364 371 327 331 359 347 356 339 319 317 335 346 329 315 324 322 325 332
 336 337 330 316 313 306 307 328 310 340 308 318 334 301 289 302 320 323
 326 311 297 314 304 292 305 312 294 287 300 299 285 303 288 278 296 277
 309 291 283 286 293 295 298 276 282 272 284 290 271 275 279 281 262 263
 280 268 270 269 264 273 265 252 267 257 274 266 259 247 261 251 233 239
 253 258 254 260 244 240 255 256 250 238 243 249 248 245 241 232]
column Name: A/W unique values: ['Medium' 'High' 'Low']
column Name: D/W unique values: ['Low' 'Medium' 'High' nan]
column Name: PAC unique values: [85. 89. 87. 76. 91. 78. 93. 86. 96. 88. 94. 65. 84. 74. 71. 77. 68. 75.
 54. 79. 83. 80. 81. 82. 63. 67. 90. 66. 42. 73. 70. 64. 57. 58. 69. 72.
 50. 59. 92. 60. 62. 55. 52. 56. 61. 53. 45. nan 37. 95. 43. 44. 46. 48.
 49. 47. 34. 39. 40. 51. 41. 36. 32. 33. 30. 31. 38. 35. 28. 29. 25.]
column Name: SHO unique values: [92. 93. 86. 85. 91. 88. 60. 73. 89. 87. 70. 90. 81. 66. 72. 82. 28. 74.
 77. 62. 50. 83. 69. 80. 46. 76. 54. 49. 61. 58. 79. 68. 59. 41. 45. 64.
 78. 55. 75. 65. 63. 48. 42. 56. 51. 30. 47. 84. 40. 57. 25. 71. 37. 43.
 53. 67. 38. 52. 39. 35. 36. 44. 32. nan 34. 33. 31. 27. 22. 29. 26. 23.
 18. 24. 20. 16. 21. 19. 17.]
column Name: PAS unique values: [91. 81. 78. 93. 86. 85. 88. 71. 80. 76. 74. 77. 79. 84. 73. 55. 83. 87.
 72. 75. 58. 89. 82. 68. 67. 64. 66. 59. 69. 90. 65. 53. 63. 62. 70. 56.
 42. 54. 61. 57. nan 60. 48. 52. 47. 46. 44. 45. 50. 51. 49. 43. 36. 38.
 40. 41. 35. 39. 34. 33. 37. 30. 32. 29. 31. 26. 28. 25. 27.]
column Name: DRI unique values: [95. 89. 90. 88. 94. 85. 91. 71. 72. 86. 73. 81. 84. 92. 80. 68. 77. 87.
 60. 83. 78. 64. 67. 79. 69. 66. 65. 70. 82. 75. 61. 74. 54. 76. 49. 63.
 59. 62. 56. nan 55. 50. 57. 58. 52. 53. 51. 48. 47. 46. 39. 44. 43. 36.
 40. 45. 41. 37. 34. 35. 42. 32. 38. 31. 33. 30. 29. 28. 25. 27.]
column Name: DEF unique values: [38. 35. 52. 64. 36. 43. 45. 51. 39. 91. 44. 86. 48. 57. 40. 88. 33. 81.
 63. 47. 53. 89. 71. 37. 80. 68. 85. 61. 90. 83. 49. 56. 58. 82. 87. 79.
 66. 55. 78. 32. 50. 76. 77. 70. 75. 41. 29. 73. 65. 59. 84. 54. 72. 46.
 42. 69. 34. 31. 30. 74. 24. 62. 25. 20. nan 26. 60. 27. 23. 28. 67. 22.
 19. 18. 21. 17. 15. 16. 12.]
```

```
column Name: PHY unique values: [65. 77. 90. 78. 59. 82. 75. 91. 76. 88. 86. 85. 73. 67. 79. 63. 83. 89.
 66. 69. 72. 64. 71. 81. 87. 68. 84. 80. 55. 70. 44. 62. 51. 57. 60. 58.
 56. 74. 52. 61. 53. 45. 50. nan 54. 47. 48. 49. 42. 37. 40. 39. 43. 38.
 46. 41. 34. 35. 36. 31. 32. 33. 29. 28.]
column Name: playerName unique values: ['lionelmessi' 'cronaldodossantosaveiro' 'janoblak' ... 'ronanmckinley'
 'zhenaowang' 'xiaozhou']
column Name: playerStatus unique values: ['Active' 'On Loan']
column Name: LB unique values: [0 1]
column Name: RW unique values: [1 0]
column Name: CAM unique values: [0 1]
column Name: CM unique values: [0 1]
column Name: CF unique values: [1 0]
column Name: CB unique values: [0 1]
column Name: RM unique values: [0 1]
column Name: GK unique values: [0 1]
column Name: RWB unique values: [0 1]
column Name: LWB unique values: [0 1]
column Name: CDM unique values: [0 1]
column Name: RB unique values: [0 1]
column Name: ST unique values: [1 0]
column Name: LM unique values: [0 1]
column Name: LW unique values: [0 1]
column Name: W/F1 unique values: [4 3 5 2 1]
column Name: SM1 unique values: [4 5 1 2 3]
column Name: IR1 unique values: [5 3 4 2 1]
column Name: weight unique values: [ 72.       83.       87.       70.       68.       80.       71.
  91.       73.       85.       92.       69.       84.       96.
  81.       82.       75.       86.       89.       74.       76.
  64.       78.       90.       66.       60.       94.       79.
  67.       65.       59.       61.       93.       88.       97.
  77.       62.       63.       95.      100.            nan  58.
  83.00697  81.19261  78.01748  88.90364  79.83184  83.91415  77.1103
  92.07877  76.20312  73.02799  66.22414  58.9667   86.1821   78.92466
  67.13132  74.84235  72.12081  87.08928  82.09979  63.04901  69.85286
  71.21363  73.93517  98.      103.       99.      102.       56.
 101.       57.       55.      104.      107.      110.       53.
  50.       54.       52.     ]
column Name: height unique values: [170.   187.   188.   181.   175.   184.   191.   178.   193.   185.
 199.   173.   168.   176.   177.   183.   180.   189.   179.   195.
 172.   182.   186.   192.   165.   194.   167.   196.   163.   190.
 174.   169.   171.   197.   200.   166.   187.96 164.   198.   190.5
 195.58 154.94 193.04 185.42 182.88 152.4  175.26 167.64 170.18 162.56
 201.   158.   162.   161.   160.   203.   157.   156.   202.   159.
 206.   155.  ]
column Name: value unique values: [1.035e+08 6.300e+07 1.200e+08 1.290e+08 1.320e+08 1.110e+08 1.205e+08
```

```
 1.020e+08 1.855e+08 1.100e+08 1.130e+08 9.050e+07 8.200e+07 1.750e+07
 8.350e+07 3.350e+07 1.145e+08 7.800e+07 1.030e+08 1.090e+08 9.200e+07
 1.000e+07 7.650e+07 8.950e+07 8.750e+07 7.950e+07 1.240e+08 1.140e+08
 9.500e+07 9.250e+07 1.055e+08 8.850e+07 8.500e+07 8.150e+07 2.600e+07
 2.100e+07 5.600e+07 6.750e+07 5.300e+07 3.650e+07 5.100e+07 6.550e+07
 4.650e+07 6.150e+07 7.250e+07 7.750e+07 4.350e+07 3.250e+07 3.600e+07
 3.200e+07 5.400e+07 4.950e+07 5.700e+07 6.650e+07 7.450e+07 7.150e+07
 1.210e+08 9.900e+07 6.700e+07 8.650e+07 9.350e+07 7.000e+07 6.200e+07
 6.600e+07 5.800e+07 4.400e+07 8.100e+07 3.700e+07 1.450e+07 4.600e+07
 4.750e+07 5.250e+07 5.450e+07 3.450e+07 5.750e+07 5.150e+07 4.450e+07
 5.500e+07 4.800e+07 6.050e+07 6.350e+07 6.100e+07 2.900e+07 5.850e+07
 5.550e+07 4.200e+07 4.050e+07 4.300e+07 4.550e+07 3.400e+07 2.650e+07
 4.250e+07 3.550e+07 4.500e+07 4.150e+07 4.000e+07 1.100e+07 1.350e+07
 2.950e+07 2.700e+07 1.550e+07 3.850e+07 5.200e+07 3.300e+07 1.900e+07
 7.350e+07 3.800e+07 3.500e+07 4.700e+07 2.400e+07 3.050e+07 1.800e+07
 2.800e+07 2.550e+07 2.500e+07 3.100e+07 2.350e+07 3.000e+07 3.150e+07
 2.250e+07 2.850e+07 4.000e+06 1.250e+07 3.750e+07 2.750e+07 1.600e+07
 1.500e+07 2.050e+07 2.200e+07 3.400e+06 5.000e+06 5.650e+07 6.250e+07
 0.000e+00 3.900e+07 2.450e+07 2.150e+07 1.300e+07 8.000e+06 2.000e+07
 8.500e+06 2.900e+06 9.000e+06 4.600e+06 5.000e+07 2.300e+07 1.850e+07
 7.000e+06 1.950e+07 5.500e+06 7.500e+06 3.800e+06 1.400e+07 1.050e+07
 1.650e+07 3.600e+06 9.500e+06 3.950e+07 1.700e+07 1.200e+07 1.150e+07
 4.900e+06 3.000e+06 1.900e+06 6.500e+06 1.700e+06 2.400e+06 3.100e+06
 6.000e+06 3.700e+06 4.700e+06 4.300e+06 2.100e+06 1.200e+06 1.800e+06
 4.800e+06 3.200e+06 1.300e+06 8.250e+05 2.300e+06 1.500e+06 3.900e+06
 2.600e+06 3.500e+06 2.800e+06 2.700e+06 4.400e+06 4.100e+06 9.500e+05
 1.600e+06 6.250e+05 1.100e+06 4.500e+06 4.200e+06 2.200e+06 3.300e+06
 1.400e+06 2.000e+06 4.750e+05 9.250e+05 7.500e+05 7.250e+05 2.500e+06
 1.000e+06 3.500e+05 5.250e+05 6.000e+05 8.500e+05 8.000e+05 5.500e+05
 2.500e+05 4.000e+05 4.250e+05 5.750e+05 2.100e+05 3.250e+05 9.000e+05
 8.750e+05 6.500e+05 7.000e+05 5.000e+05 9.750e+05 3.750e+05 7.750e+05
 2.750e+05 1.800e+05 4.500e+05 6.750e+05 1.500e+05 2.400e+05 3.000e+05
 1.300e+05 2.200e+05 2.000e+05 1.100e+05 1.700e+05 2.300e+05 9.000e+04
 1.200e+05 8.000e+04 1.900e+05 1.400e+05 1.600e+05 1.000e+05 6.000e+04
 5.000e+04 7.000e+04 4.500e+04 3.500e+04 4.000e+04 2.500e+04 2.000e+04
 1.500e+04 3.000e+04 9.000e+03]
column Name: wage unique values: [560000. 220000. 125000. 370000. 270000. 240000. 250000. 160000. 260000.
 210000. 310000. 130000. 350000. 300000. 190000. 145000. 195000. 100000.
 140000. 290000.  82000. 110000. 230000. 155000. 200000. 165000.  95000.
 170000. 105000. 115000. 150000. 135000.  55000.  58000.  81000.  34000.
 120000.  59000.  90000.  65000.  56000.  71000.  18000.  75000.  47000.
  20000.  84000.  86000.  74000.  78000.  27000.  68000.  85000.  25000.
  46000.  83000.  54000.  79000. 175000.  43000.  49000.  45000.  38000.
  41000.  39000.  23000.  51000.  50000.  87000.  30000.  14000.  69000.
  31000.  64000.  53000.  35000.  21000.  28000.  17000.  33000.  70000.
  32000.  89000.  26000.  40000.  76000.  72000.  48000.  36000.  29000.
  60000.  16000.  37000.  24000.  52000.      0.  62000.  73000.  63000.
  19000.   1000.  66000.  80000.  12000.   2000.  42000.  13000. 900000.
  57000.  77000.  61000.  22000.  67000.  44000.  15000.  11000.   8000.
 850000.  10000.  88000. 500000.   7000.   6000.   9000.   5000. 700000.
 950000. 750000.   3000. 650000. 600000.   4000. 800000. 550000.]
column Name: Release clause unique values: [1.384e+08 7.590e+07 1.594e+08 ... 5.900e+04 3.500e+04 6.400e+04]
column Name: height_bins unique values: [(160, 170], (180, 190], (170, 180], (190, 200], (150, 160], (200, 210]]
Categories (20, interval[int64, right]): [(10, 20] < (20, 30] < (30, 40] < (40, 50] ... (170, 180] < (180, 190] < (190,
200] < (200, 210]]
column Name: weight_bins unique values: [(70.0, 80.0], (80.0, 90.0], (60.0, 70.0], (90.0, 100.0], (50.0, 60.0], NaN, (1
00.0, 110.0], (40.0, 50.0]]
Categories (10, interval[int64, right]): [(10, 20] < (20, 30] < (30, 40] < (40, 50] ... (70, 80] < (80, 90] < (90, 100]
< (100, 110]]
column Name: wage_bins unique values: [[550000, 600000), [200000, 250000), [100000, 150000), [350000, 400000), [250000,
300000), ..., [950000, 960000), [750000, 800000), [650000, 700000), [600000, 650000), [800000, 850000)]
Length: 18
Categories (20, interval[int64, left]): [[0, 50000) < [50000, 100000) < [100000, 150000) < [150000, 200000) ... [80000
0, 850000) < [850000, 900000) < [900000, 950000) < [950000, 960000)]
column Name: value_bins unique values: [[100000000, 150000000), [50000000, 100000000), [150000000, 200000000), [0, 5000
0000)]
Categories (4, interval[int64, left]): [[0, 50000000) < [50000000, 100000000) < [100000000, 150000000) < [150000000, 20
0000000)]
column Name: Release clause_bins unique values: [[100000000.0, 150000000.0), [50000000.0, 100000000.0), [150000000.0, 2
00000000.0), [200000000.0, 250000000.0), [0.0, 50000000.0), NaN]
Categories (5, interval[int64, left]): [[0, 50000000) < [50000000, 100000000) < [100000000, 150000000) < [150000000, 20
0000000) < [200000000, 250000000)]
```

In [ ]:

In [71]:
```
df_unClean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19021 entries, 0 to 19020
Data columns (total 88 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   HITS                16426 non-null  float64
 1   Age                 19021 non-null  int64  
 2   ↓OVA                19019 non-null  float64
 3   POT                 19020 non-null  float64
 4   Preferred Foot      19021 non-null  object 
 5   BOV                 19021 non-null  int64  
 6   Best Position       19021 non-null  object 
 7   Attacking           19020 non-null  float64
 8   Crossing            19020 non-null  float64
 9   Finishing           19016 non-null  float64
 10  Heading Accuracy    19013 non-null  float64
 11  Short Passing       19012 non-null  object 
 12  Volleys             19014 non-null  float64
 13  Skill               19015 non-null  float64
 14  Dribbling           19020 non-null  object 
 15  Curve               19013 non-null  float64
 16  FK Accuracy         19015 non-null  float64
 17  Long Passing        19018 non-null  float64
 18  Ball Control        19018 non-null  float64
 19  Movement            19016 non-null  float64
 20  Acceleration        19017 non-null  float64
 21  Sprint Speed        19018 non-null  float64
 22  Agility             19019 non-null  float64
 23  Reactions           19017 non-null  float64
 24  Balance             19014 non-null  float64
 25  Power               19020 non-null  float64
 26  Shot Power          19019 non-null  float64
 27  Jumping             19017 non-null  float64
 28  Stamina             19020 non-null  float64
 29  Strength            19016 non-null  float64
 30  Long Shots          19014 non-null  float64
 31  Mentality           19015 non-null  float64
 32  Aggression          19021 non-null  int64  
 33  Interceptions       19017 non-null  float64
 34  Positioning         19020 non-null  float64
 35  Vision              19021 non-null  int64  
 36  Penalties           19020 non-null  float64
 37  Composure           19020 non-null  float64
 38  Defending           19020 non-null  float64
 39  Marking             19019 non-null  float64
 40  Standing Tackle      19018 non-null  float64
 41  Sliding Tackle       19020 non-null  float64
 42  Goalkeeping          19018 non-null  float64
 43  GK Diving            19020 non-null  float64
 44  GK Handling          19020 non-null  float64
 45  GK Kicking           19019 non-null  float64
 46  GK Positioning       19019 non-null  float64
 47  GK Reflexes          19021 non-null  int64
 48  Total Stats          19020 non-null  float64
 49  Base Stats           19021 non-null  int64
 50  A/W                  19021 non-null  object
 51  D/W                  19020 non-null  object
 52  PAC                  19018 non-null  float64
 53  SHO                  19018 non-null  float64
 54  PAS                  19016 non-null  float64
 55  DRI                  19019 non-null  float64
 56  DEF                  19016 non-null  float64
 57  PHY                  19020 non-null  float64
 58  playerName           19021 non-null  object
 59  playerStatus         19021 non-null  object
 60  LB                   19021 non-null  int64
 61  RW                   19021 non-null  int64
 62  CAM                  19021 non-null  int64
 63  CM                   19021 non-null  int64
 64  CF                   19021 non-null  int64
 65  CB                   19021 non-null  int64
 66  RM                   19021 non-null  int64
 67  GK                   19021 non-null  int64
 68  RWB                  19021 non-null  int64
 69  LWB                  19021 non-null  int64
 70  CDM                  19021 non-null  int64
 71  RB                   19021 non-null  int64
 72  ST                   19021 non-null  int64
 73  LM                   19021 non-null  int64
 74  LW                   19021 non-null  int64
 75  W/F1                 19021 non-null  int64
 76  SM1                  19021 non-null  int64
 77  IR1                  19021 non-null  int64
 78  weight               19020 non-null  float64
 79  height               19021 non-null  float64
 80  value                19021 non-null  float64
 81  wage                 19021 non-null  float64
 82  Release clause       19018 non-null  float64
 83  height_bins          19021 non-null  category
 84  weight_bins          19020 non-null  category
 85   wage_bins            19021 non-null   category
 86   value_bins           19021 non-null   category
 87   Release clause_bins  19018 non-null   category
dtypes: category(5), float64(51), int64(24), object(8)
memory usage: 12.1+ MB
```

In [72]:
```
# conclusion about my data using df_unclean.info()
# 1.Data contains 88 features and 19021 samples
# 2. There are missing data in the features
# 3. The data is a mixture of numeric and catgorical variables
# 4. Some features have a data type mismatch
```

In [ ]:

In [73]:
```
# checking for the data types of each column

df_unClean.dtypes
```

Out[73]:
```
HITS                    float64
Age                       int64
↓OVA                    float64
POT                     float64
Preferred Foot           object
                         ...
height_bins            category
weight_bins            category
wage_bins              category
value_bins             category
Release clause_bins    category
Length: 88, dtype: object
```

In [ ]:

In [74]:
```
# After visual inspection of unique values and data types in the new dataframe
# I observed that two features where represented as object data types due to the presence of a string
# meaning dribbling and short pass are data type mismatch
# Converting Short Passing feature to float
shrtList = []
for val in df_unClean['Short Passing']:
    if pd.isna(val):
        shrtList.append(val)
    else:
        val = str(val)
        if '_' in val:
            val = float(val[:-1])
            shrtList.append(val)
        else:
            val = float(val[:])
            shrtList.append(val)
df_unClean['shortPass'] = shrtList
```

In [ ]:

In [75]:
```
# Converting Dribbling feature to int
dribList = []
for val in df_unClean['Dribbling']:
    if pd.isna(val):
        dribList.append(val)
    else:
        val = str(val)
        if '_' in val:
            val = int(val[:-1])
            dribList.append(val)
        else:
            val = int(val[:])
            dribList.append(val)
df_unClean['dribbling'] = dribList
```

In [ ]:

In [76]:
```
# Dropping the old column that had data type mismatch  (Short Passing and Dribbling)
df_unClean = df_unClean.drop (['Short Passing','Dribbling'],axis = 1)
```

In [ ]:

In [77]:
```
df_unClean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19021 entries, 0 to 19020
Data columns (total 88 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   HITS                  16426 non-null  float64
 1   Age                   19021 non-null  int64  
 2   ↓OVA                  19019 non-null  float64
 3   POT                   19020 non-null  float64
 4   Preferred Foot        19021 non-null  object 
 5   BOV                   19021 non-null  int64  
 6   Best Position         19021 non-null  object 
 7   Attacking             19020 non-null  float64
 8   Crossing              19020 non-null  float64
 9   Finishing             19016 non-null  float64
 10  Heading Accuracy      19013 non-null  float64
 11  Volleys               19014 non-null  float64
 12  Skill                 19015 non-null  float64
 13  Curve                 19013 non-null  float64
 14  FK Accuracy           19015 non-null  float64
 15  Long Passing          19018 non-null  float64
 16  Ball Control          19018 non-null  float64
 17  Movement              19016 non-null  float64
 18  Acceleration          19017 non-null  float64
 19  Sprint Speed          19018 non-null  float64
 20  Agility               19019 non-null  float64
 21  Reactions             19017 non-null  float64
 22  Balance               19014 non-null  float64
 23  Power                 19020 non-null  float64
 24  Shot Power            19019 non-null  float64
 25  Jumping               19017 non-null  float64
 26  Stamina               19020 non-null  float64
 27  Strength              19016 non-null  float64
 28  Long Shots            19014 non-null  float64
 29  Mentality             19015 non-null  float64
 30  Aggression            19021 non-null  int64  
 31  Interceptions         19017 non-null  float64
 32  Positioning           19020 non-null  float64
 33  Vision                19021 non-null  int64  
 34  Penalties             19020 non-null  float64
 35  Composure             19020 non-null  float64
 36  Defending             19020 non-null  float64
 37  Marking               19019 non-null  float64
 38  Standing Tackle       19018 non-null  float64
 39  Sliding Tackle        19020 non-null  float64
 40  Goalkeeping          19018 non-null  float64
 41  GK Diving            19020 non-null  float64
 42  GK Handling          19020 non-null  float64
 43  GK Kicking           19019 non-null  float64
 44  GK Positioning       19019 non-null  float64
 45  GK Reflexes          19021 non-null  int64
 46  Total Stats          19020 non-null  float64
 47  Base Stats           19021 non-null  int64
 48  A/W                  19021 non-null  object
 49  D/W                  19020 non-null  object
 50  PAC                  19018 non-null  float64
 51  SHO                  19018 non-null  float64
 52  PAS                  19016 non-null  float64
 53  DRI                  19019 non-null  float64
 54  DEF                  19016 non-null  float64
 55  PHY                  19020 non-null  float64
 56  playerName           19021 non-null  object
 57  playerStatus         19021 non-null  object
 58  LB                   19021 non-null  int64
 59  RW                   19021 non-null  int64
 60  CAM                  19021 non-null  int64
 61  CM                   19021 non-null  int64
 62  CF                   19021 non-null  int64
 63  CB                   19021 non-null  int64
 64  RM                   19021 non-null  int64
 65  GK                   19021 non-null  int64
 66  RWB                  19021 non-null  int64
 67  LWB                  19021 non-null  int64
 68  CDM                  19021 non-null  int64
 69  RB                   19021 non-null  int64
 70  ST                   19021 non-null  int64
 71  LM                   19021 non-null  int64
 72  LW                   19021 non-null  int64
 73  W/F1                 19021 non-null  int64
 74  SM1                  19021 non-null  int64
 75  IR1                  19021 non-null  int64
 76  weight               19020 non-null  float64
 77  height               19021 non-null  float64
 78  value                19021 non-null  float64
 79  wage                 19021 non-null  float64
 80  Release clause       19018 non-null  float64
 81  height_bins          19021 non-null  category
 82  weight_bins          19020 non-null  category
 83  wage_bins            19021 non-null  category
 84  value_bins           19021 non-null  category
 85  Release clause_bins  19018 non-null  category
 86  shortPass            19012 non-null  float64
 87  dribbling            19020 non-null  float64
dtypes: category(5), float64(53), int64(24), object(6)
memory usage: 12.1+ MB
```

In [ ]:

```

```

In [78]:

```
# CHECKING FOR MISSING DATA

# Inspecting for missing values
misscol_counts = {}
for cols in df_unClean.columns:
    misscol_counts[cols] = df_unClean[cols].isna().sum()
for col,count in misscol_counts.items():
    if count > 0:
        print(f'Column Name : {col} \nMissing Values : {count}')
```

```
Column Name : HITS
Missing Values : 2595
Column Name : ↓OVA
Missing Values : 2
Column Name : POT
Missing Values : 1
Column Name : Attacking
Missing Values : 1
Column Name : Crossing
Missing Values : 1
Column Name : Finishing
Missing Values : 5
Column Name : Heading Accuracy
Missing Values : 8
Column Name : Volleys
Missing Values : 7
Column Name : Skill
Missing Values : 6
Column Name : Curve
Missing Values : 8
Column Name : FK Accuracy
Missing Values : 6
Column Name : Long Passing
Missing Values : 3
Column Name : Ball Control
Missing Values : 3
Column Name : Movement
Missing Values : 5
Column Name : Acceleration
Missing Values : 4
Column Name : Sprint Speed
Missing Values : 3
Column Name : Agility
Missing Values : 2
Column Name : Reactions
Missing Values : 4
Column Name : Balance
Missing Values : 7
Column Name : Power
Missing Values : 1
Column Name : Shot Power
Missing Values : 2
Column Name : Jumping
Missing Values : 4
Column Name : Stamina
```

```
Missing Values : 1
Column Name : Strength
Missing Values : 5
Column Name : Long Shots
Missing Values : 7
Column Name : Mentality
Missing Values : 6
Column Name : Interceptions
Missing Values : 4
Column Name : Positioning
Missing Values : 1
Column Name : Penalties
Missing Values : 1
Column Name : Composure
Missing Values : 1
Column Name : Defending
Missing Values : 1
Column Name : Marking
Missing Values : 2
Column Name : Standing Tackle
Missing Values : 3
Column Name : Sliding Tackle
Missing Values : 1
Column Name : Goalkeeping
Missing Values : 3
Column Name : GK Diving
Missing Values : 1
Column Name : GK Handling
Missing Values : 1
Column Name : GK Kicking
Missing Values : 2
Column Name : GK Positioning
Missing Values : 2
Column Name : Total Stats
Missing Values : 1
Column Name : D/W
Missing Values : 1
Column Name : PAC
Missing Values : 3
Column Name : SHO
Missing Values : 3
Column Name : PAS
Missing Values : 5
Column Name : DRI
Missing Values : 2
```

```
Column Name : DEF
Missing Values : 5
Column Name : PHY
Missing Values : 1
Column Name : weight
Missing Values : 1
Column Name : Release clause
Missing Values : 3
Column Name : weight_bins
Missing Values : 1
Column Name : Release clause_bins
Missing Values : 3
Column Name : shortPass
Missing Values : 9
Column Name : dribbling
Missing Values : 1
```

In [ ]:

In [79]:

```
# BEFORE TREATING FOR MISSING VALUE, I WILL MAKE A COPY OF MY DATASET
# MAKING A COPY OF MY DATASET.

df_c = df_unClean.copy()
```

In [ ]:

In [80]:

```
df_c
```

Out[80]:

	HITS	Age	↓OVA	POT	Preferred Foot	BOV	Best Position	Attacking	Crossing	Finishing	...	value	wage	Release clause	height_bins
0	771.0	33	93.0	93.0	Left	93	RW	429.0	85.0	95.0	...	103500000.0	560000.0	138400000.0	(160, 170]
1	562.0	35	92.0	92.0	Right	92	ST	437.0	84.0	95.0	...	63000000.0	220000.0	75900000.0	(180, 190]
2	150.0	27	91.0	93.0	Right	91	GK	95.0	13.0	11.0	...	120000000.0	125000.0	159400000.0	(180, 190]
3	207.0	29	91.0	91.0	Right	91	CAM	407.0	94.0	82.0	...	129000000.0	370000.0	161000000.0	(180, 190]
4	595.0	28	91.0	91.0	Right	91	LW	408.0	85.0	87.0	...	132000000.0	270000.0	166500000.0	(170, 180]
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
19016	NaN	21	47.0	55.0	Right	49	CB	145.0	23.0	26.0	...	100000.0	1000.0	70000.0	(170, 180]
19017	NaN	17	47.0	67.0	Right	51	CAM	211.0	38.0	42.0	...	130000.0	500000.0	165000.0	(170, 180]
19018	NaN	18	47.0	65.0	Right	49	CAM	200.0	30.0	34.0	...	120000.0	500000.0	131000.0	(170, 180]
19019	NaN	20	47.0	57.0	Right	48	ST	215.0	45.0	52.0	...	100000.0	2000.0	88000.0	(170, 180]
19020	NaN	21	47.0	57.0	Left	50	LB	163.0	40.0	18.0	...	100000.0	1000.0	79000.0	(180, 190]

19021 rows × 88 columns

In [ ]:

```
```

In [81]:

```
# Treating for missing values
# Dropping all rows with misssing values for the target variable hits, and assigning it to a variable test_1

test_data = df_c[df_c['HITS'].isna()]
```

```
In [82]: test_data
```

Out[82]:

	HITS	Age	↓OVA	POT	Preferred Foot	BOV	Best Position	Attacking	Crossing	Finishing	...	value	wage	Release clause	height_bins	weigl
16245	NaN	25	58.0	62.0	Left	60	CB	233.0	51.0	42.0	...	250000.0	500000.0	244000.0	(170, 180]	
16246	NaN	21	58.0	70.0	Right	61	RM	237.0	54.0	46.0	...	500000.0	750000.0	334000.0	(170, 180]	
16247	NaN	27	58.0	59.0	Right	59	ST	256.0	32.0	59.0	...	240000.0	650000.0	229000.0	(190, 200]	
16248	NaN	23	58.0	64.0	Left	58	LB	191.0	53.0	26.0	...	300000.0	2000.0	218000.0	(170, 180]	
16250	NaN	30	58.0	58.0	Right	58	CB	178.0	27.0	22.0	...	170000.0	900000.0	135000.0	(180, 190]	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
19016	NaN	21	47.0	55.0	Right	49	CB	145.0	23.0	26.0	...	100000.0	1000.0	70000.0	(170, 180]	
19017	NaN	17	47.0	67.0	Right	51	CAM	211.0	38.0	42.0	...	130000.0	500000.0	165000.0	(170, 180]	
19018	NaN	18	47.0	65.0	Right	49	CAM	200.0	30.0	34.0	...	120000.0	500000.0	131000.0	(170, 180]	
19019	NaN	20	47.0	57.0	Right	48	ST	215.0	45.0	52.0	...	100000.0	2000.0	88000.0	(170, 180]	
19020	NaN	21	47.0	57.0	Left	50	LB	163.0	40.0	18.0	...	100000.0	1000.0	79000.0	(180, 190]	

2595 rows × 88 columns

```
In [ ]: 
```

```
In [83]: df_c
```

Out[83]:

	HITS	Age	↓OVA	POT	Preferred Foot	BOV	Best Position	Attacking	Crossing	Finishing	...	value	wage	Release clause	height_bins
0	771.0	33	93.0	93.0	Left	93	RW	429.0	85.0	95.0	...	103500000.0	560000.0	138400000.0	(160, 170]
1	562.0	35	92.0	92.0	Right	92	ST	437.0	84.0	95.0	...	63000000.0	220000.0	75900000.0	(180, 190]
2	150.0	27	91.0	93.0	Right	91	GK	95.0	13.0	11.0	...	120000000.0	125000.0	159400000.0	(180, 190]
3	207.0	29	91.0	91.0	Right	91	CAM	407.0	94.0	82.0	...	129000000.0	370000.0	161000000.0	(180, 190]
4	595.0	28	91.0	91.0	Right	91	LW	408.0	85.0	87.0	...	132000000.0	270000.0	166500000.0	(170, 180]
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
19016	NaN	21	47.0	55.0	Right	49	CB	145.0	23.0	26.0	...	100000.0	1000.0	70000.0	(170, 180]
19017	NaN	17	47.0	67.0	Right	51	CAM	211.0	38.0	42.0	...	130000.0	500000.0	165000.0	(170, 180]
19018	NaN	18	47.0	65.0	Right	49	CAM	200.0	30.0	34.0	...	120000.0	500000.0	131000.0	(170, 180]
19019	NaN	20	47.0	57.0	Right	48	ST	215.0	45.0	52.0	...	100000.0	2000.0	88000.0	(170, 180]
19020	NaN	21	47.0	57.0	Left	50	LB	163.0	40.0	18.0	...	100000.0	1000.0	79000.0	(180, 190]

19021 rows × 88 columns

In [ ]:

In [84]:

```
# Dropping rows with misssing values for the target variable hits
# (Since there is no ground truth for the model to learn from for those instances.)

df_c = df_c.dropna(subset = 'HITS')
```

```
In [85]: df_c
```

Out[85]:

	HITS	Age	↓OVA	POT	Preferred Foot	BOV	Best Position	Attacking	Crossing	Finishing	...	value	wage	Release clause	height_bins
0	771.0	33	93.0	93.0	Left	93	RW	429.0	85.0	95.0	...	103500000.0	560000.0	138400000.0	(160, 170]
1	562.0	35	92.0	92.0	Right	92	ST	437.0	84.0	95.0	...	63000000.0	220000.0	75900000.0	(180, 190]
2	150.0	27	91.0	93.0	Right	91	GK	95.0	13.0	11.0	...	120000000.0	125000.0	159400000.0	(180, 190]
3	207.0	29	91.0	91.0	Right	91	CAM	407.0	94.0	82.0	...	129000000.0	370000.0	161000000.0	(180, 190]
4	595.0	28	91.0	91.0	Right	91	LW	408.0	85.0	87.0	...	132000000.0	270000.0	166500000.0	(170, 180]
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
16658	1.0	19	58.0	73.0	Right	58	RB	208.0	55.0	31.0	...	475000.0	550000.0	494000.0	(170, 180]
16659	1.0	20	58.0	70.0	Right	58	LB	213.0	51.0	28.0	...	475000.0	950000.0	420000.0	(170, 180]
16660	1.0	25	58.0	62.0	Right	60	CM	232.0	35.0	48.0	...	275000.0	2000.0	292000.0	(170, 180]
16661	1.0	34	58.0	58.0	Left	58	CB	172.0	27.0	15.0	...	70000.0	550000.0	56000.0	(180, 190]
16678	6.0	22	58.0	67.0	Right	60	CB	166.0	35.0	16.0	...	425000.0	3000.0	414000.0	(180, 190]

16426 rows × 88 columns

```
In [ ]:
```

```
In [86]: # Dropping the rest of the rows with missing values in our datasets due to Sample size
         # (Number of nan is relatively small compared to the size of the dataset)
```

```
df_c = df_c.dropna()
```

In [87]:
```
df_c
```

Out[87]:

	HITS	Age	↓OVA	POT	Preferred Foot	BOV	Best Position	Attacking	Crossing	Finishing	...	value	wage	Release clause	height_bins
0	771.0	33	93.0	93.0	Left	93	RW	429.0	85.0	95.0	...	103500000.0	560000.0	138400000.0	(160, 170]
1	562.0	35	92.0	92.0	Right	92	ST	437.0	84.0	95.0	...	63000000.0	220000.0	75900000.0	(180, 190]
2	150.0	27	91.0	93.0	Right	91	GK	95.0	13.0	11.0	...	120000000.0	125000.0	159400000.0	(180, 190]
3	207.0	29	91.0	91.0	Right	91	CAM	407.0	94.0	82.0	...	129000000.0	370000.0	161000000.0	(180, 190]
4	595.0	28	91.0	91.0	Right	91	LW	408.0	85.0	87.0	...	132000000.0	270000.0	166500000.0	(170, 180]
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
16658	1.0	19	58.0	73.0	Right	58	RB	208.0	55.0	31.0	...	475000.0	550000.0	494000.0	(170, 180]
16659	1.0	20	58.0	70.0	Right	58	LB	213.0	51.0	28.0	...	475000.0	950000.0	420000.0	(170, 180]
16660	1.0	25	58.0	62.0	Right	60	CM	232.0	35.0	48.0	...	275000.0	2000.0	292000.0	(170, 180]
16661	1.0	34	58.0	58.0	Left	58	CB	172.0	27.0	15.0	...	70000.0	550000.0	56000.0	(180, 190]
16678	6.0	22	58.0	67.0	Right	60	CB	166.0	35.0	16.0	...	425000.0	3000.0	414000.0	(180, 190]

16326 rows × 88 columns

In [88]:
```
# making sure there are no missing data.
df_c.isna().sum()
```

```
Out[88]: HITS                   0
         Age                    0
         ↓OVA                   0
         POT                    0
         Preferred Foot         0
                               ..
         wage_bins              0
         value_bins             0
         Release clause_bins    0
         shortPass              0
         dribbling              0
         Length: 88, dtype: int64
```

```
In [ ]:
```

```
In [89]: # checking the size of the data
         df_c.shape

Out[89]: (16326, 88)
```

```
In [90]: # checking the lenght of data lost.
         # thats establishing the quantity
         len(df_c)-len(df_unClean)

Out[90]: -2695
```

```
In [ ]:
```

```
In [91]: # checking for duplicates
         # the reason for removing duplicates is cause i don"t want my model to do double counting,


         print('Duplicate in Dataset:', df_c.duplicated().sum())

         Duplicate in Dataset: 42
```

```
In [92]: # dropping duplicates
         df_c = df_c.drop_duplicates()
```

```
In [93]: df_c.duplicated().sum()

Out[93]: 0
```

```
In [94]: len(df_c)

Out[94]: 16284

In [95]: # arranging the index
         df_c.reset_index(inplace =True, drop= True)

In [ ]:

In [96]: df_c
```

```
Out[96]:
```

	HITS	Age	↓OVA	POT	Preferred Foot	BOV	Best Position	Attacking	Crossing	Finishing	...	value	wage	Release clause	height_bins
0	771.0	33	93.0	93.0	Left	93	RW	429.0	85.0	95.0	...	103500000.0	560000.0	138400000.0	(160, 170]
1	562.0	35	92.0	92.0	Right	92	ST	437.0	84.0	95.0	...	63000000.0	220000.0	75900000.0	(180, 190]
2	150.0	27	91.0	93.0	Right	91	GK	95.0	13.0	11.0	...	120000000.0	125000.0	159400000.0	(180, 190]
3	207.0	29	91.0	91.0	Right	91	CAM	407.0	94.0	82.0	...	129000000.0	370000.0	161000000.0	(180, 190]
4	595.0	28	91.0	91.0	Right	91	LW	408.0	85.0	87.0	...	132000000.0	270000.0	166500000.0	(170, 180]
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
16279	1.0	19	58.0	73.0	Right	58	RB	208.0	55.0	31.0	...	475000.0	550000.0	494000.0	(170, 180]
16280	1.0	20	58.0	70.0	Right	58	LB	213.0	51.0	28.0	...	475000.0	950000.0	420000.0	(170, 180]
16281	1.0	25	58.0	62.0	Right	60	CM	232.0	35.0	48.0	...	275000.0	2000.0	292000.0	(170, 180]
16282	1.0	34	58.0	58.0	Left	58	CB	172.0	27.0	15.0	...	70000.0	550000.0	56000.0	(180, 190]
16283	6.0	22	58.0	67.0	Right	60	CB	166.0	35.0	16.0	...	425000.0	3000.0	414000.0	(180, 190]

16284 rows × 88 columns

```
In [ ]:
```

```
In [97]:
# TREATING OUTLIERS, SINCE I CAN'T TREAT OUTLIER ON A CATEGORICAL VARIABLE.
# BELOW IS THE CODE TO SPLIT MY DATA INTO CATEGORICAL AND NUMERICAL.
# code to split my data into categorical and numerical


continous_vars = df_c.select_dtypes(include = ['float64','int']).columns
```

```
print(continous_vars)
categorical_vars = df_c.select_dtypes(include = ['object','category']).columns
print(categorical_vars)
```

```
Index(['HITS', 'Age', '↓OVA', 'POT', 'BOV', 'Attacking', 'Crossing',
       'Finishing', 'Heading Accuracy', 'Volleys', 'Skill', 'Curve',
       'FK Accuracy', 'Long Passing', 'Ball Control', 'Movement',
       'Acceleration', 'Sprint Speed', 'Agility', 'Reactions', 'Balance',
       'Power', 'Shot Power', 'Jumping', 'Stamina', 'Strength', 'Long Shots',
       'Mentality', 'Aggression', 'Interceptions', 'Positioning', 'Vision',
       'Penalties', 'Composure', 'Defending', 'Marking', 'Standing Tackle',
       'Sliding Tackle', 'Goalkeeping', 'GK Diving', 'GK Handling',
       'GK Kicking', 'GK Positioning', 'GK Reflexes', 'Total Stats',
       'Base Stats', 'PAC', 'SHO', 'PAS', 'DRI', 'DEF', 'PHY', 'LB', 'RW',
       'CAM', 'CM', 'CF', 'CB', 'RM', 'GK', 'RWB', 'LWB', 'CDM', 'RB', 'ST',
       'LM', 'LW', 'W/F1', 'SM1', 'IR1', 'weight', 'height', 'value', 'wage',
       'Release clause', 'shortPass', 'dribbling'],
      dtype='object')
Index(['Preferred Foot', 'Best Position', 'A/W', 'D/W', 'playerName',
       'playerStatus', 'height_bins', 'weight_bins', 'wage_bins', 'value_bins',
       'Release clause_bins'],
      dtype='object')
```

In [ ]:

In [98]:
```
# creating function for detecting outliers
def outlier_lims(col):
    q3,ql = np.percentile(col,[75,25])
    iqr = q3-ql
    upper_lim = q3 + 1.5*iqr
    lower_lim = ql - 1.5*iqr
    return upper_lim, lower_lim
```

In [99]:
```
# this is a for loop that will run through each column
for col in continous_vars:
    print("---------------------------------------------")
    print('column:', col)

    UL,LL = outlier_lims(df_c[col])
    print("upper limit =", UL)# this recieve the upper and lower limit, use it to filter
    print("lower limit =", LL)

    total_outliers = len(df_c.loc[df_c[col]<LL,col]) + len(df_c.loc[df_c[col]>UL,col])#use it to filter data that is <
    percent = (total_outliers / len(df_c.index) )*100
    print ("percentage of outliers=", percent)
    print("'----------------------------------------------")
```

```
---------------------------------------------
column: HITS
upper limit = 34.5
lower limit = -17.5
percentage of outliers= 13.135593220338984
'-----------------------------------------------
---------------------------------------------
column: Age
upper limit = 39.5
lower limit = 11.5
percentage of outliers= 0.15352493244902973
'-----------------------------------------------
---------------------------------------------
column: ↓OVA
upper limit = 83.0
lower limit = 51.0
percentage of outliers= 0.8720216163104888
'-----------------------------------------------
---------------------------------------------
column: POT
upper limit = 88.0
lower limit = 56.0
percentage of outliers= 0.37460083517563253
'-----------------------------------------------
---------------------------------------------
column: BOV
upper limit = 84.0
lower limit = 52.0
percentage of outliers= 0.7860476541390321
'-----------------------------------------------
---------------------------------------------
column: Attacking
upper limit = 402.5
lower limit = 134.5
percentage of outliers= 10.249324490297223
'-----------------------------------------------
---------------------------------------------
column: Crossing
upper limit = 101.0
lower limit = 5.0
percentage of outliers= 0.0
'-----------------------------------------------
---------------------------------------------
column: Finishing
upper limit = 109.5
```

```
lower limit = -14.5
percentage of outliers= 0.0
'----------------------------------------------
--------------------------------------------
column: Heading Accuracy
upper limit = 93.5
lower limit = 17.5
percentage of outliers= 8.21665438467207
'----------------------------------------------
--------------------------------------------
column: Volleys
upper limit = 98.5
lower limit = -9.5
percentage of outliers= 0.0
'----------------------------------------------
--------------------------------------------
column: Skill
upper limit = 435.0
lower limit = 115.0
percentage of outliers= 9.217636944239745
'----------------------------------------------
--------------------------------------------
column: Curve
upper limit = 102.0
lower limit = -2.0
percentage of outliers= 0.0
'----------------------------------------------
--------------------------------------------
column: FK Accuracy
upper limit = 97.0
lower limit = -7.0
percentage of outliers= 0.0
'----------------------------------------------
--------------------------------------------
column: Long Passing
upper limit = 93.5
lower limit = 17.5
percentage of outliers= 1.2957504298698108
'----------------------------------------------
--------------------------------------------
column: Ball Control
upper limit = 88.0
lower limit = 40.0
percentage of outliers= 11.066077130926063
'----------------------------------------------
```

```
---------------------------------------------
column: Movement
upper limit = 459.0
lower limit = 195.0
percentage of outliers= 2.7143208056988453
'-----------------------------------------------
---------------------------------------------
column: Acceleration
upper limit = 100.5
lower limit = 32.5
percentage of outliers= 3.641611397690985
'-----------------------------------------------
---------------------------------------------
column: Sprint Speed
upper limit = 100.5
lower limit = 32.5
percentage of outliers= 3.2670105625153525
'-----------------------------------------------
---------------------------------------------
column: Agility
upper limit = 99.5
lower limit = 31.5
percentage of outliers= 2.536231884057971
'-----------------------------------------------
---------------------------------------------
column: Reactions
upper limit = 85.5
lower limit = 41.5
percentage of outliers= 0.7307786784573815
'-----------------------------------------------
---------------------------------------------
column: Balance
upper limit = 101.0
lower limit = 29.0
percentage of outliers= 1.2220584622942767
'-----------------------------------------------
---------------------------------------------
column: Power
upper limit = 429.5
lower limit = 185.5
percentage of outliers= 1.694915254237288
'-----------------------------------------------
---------------------------------------------
column: Shot Power
upper limit = 97.5
```

```
lower limit = 21.5
percentage of outliers= 0.1105379513633014
'-----------------------------------------------
---------------------------------------------
column: Jumping
upper limit = 95.5
lower limit = 35.5
percentage of outliers= 2.051093097519037
'-----------------------------------------------
---------------------------------------------
column: Stamina
upper limit = 96.5
lower limit = 36.5
percentage of outliers= 8.640383198231392
'-----------------------------------------------
---------------------------------------------
column: Strength
upper limit = 96.5
lower limit = 36.5
percentage of outliers= 1.9466961434536971
'-----------------------------------------------
---------------------------------------------
column: Long Shots
upper limit = 105.0
lower limit = -7.0
percentage of outliers= 0.0
'-----------------------------------------------
---------------------------------------------
column: Mentality
upper limit = 396.5
lower limit = 144.5
percentage of outliers= 9.180790960451978
'-----------------------------------------------
---------------------------------------------
column: Aggression
upper limit = 104.5
lower limit = 12.5
percentage of outliers= 0.07369196757553427
'-----------------------------------------------
---------------------------------------------
column: Interceptions
upper limit = 120.875
lower limit = -28.125
percentage of outliers= 0.0
'-----------------------------------------------
```

```
---------------------------------------------
column: Positioning
upper limit = 99.5
lower limit = 7.5
percentage of outliers= 2.4318349299926307
'-----------------------------------------------
---------------------------------------------
column: Vision
upper limit = 92.0
lower limit = 20.0
percentage of outliers= 0.6386637189879637
'-----------------------------------------------
---------------------------------------------
column: Penalties
upper limit = 92.5
lower limit = 8.5
percentage of outliers= 0.04298698108572832
'-----------------------------------------------
---------------------------------------------
column: Composure
upper limit = 89.0
lower limit = 33.0
percentage of outliers= 2.4318349299926307
'-----------------------------------------------
---------------------------------------------
column: Defending
upper limit = 354.5
lower limit = -73.5
percentage of outliers= 0.0
'-----------------------------------------------
---------------------------------------------
column: Marking
upper limit = 115.0
lower limit = -21.0
percentage of outliers= 0.0
'-----------------------------------------------
---------------------------------------------
column: Standing Tackle
upper limit = 124.0
lower limit = -28.0
percentage of outliers= 0.0
'-----------------------------------------------
---------------------------------------------
column: Sliding Tackle
upper limit = 121.0
```

```
lower limit = -31.0
percentage of outliers= 0.0
'-----------------------------------------------
---------------------------------------------
column: Goalkeeping
upper limit = 75.5
lower limit = 31.5
percentage of outliers= 10.746745271432081
'-----------------------------------------------
---------------------------------------------
column: GK Diving
upper limit = 23.0
lower limit = -1.0
percentage of outliers= 10.181773520019652
'-----------------------------------------------
---------------------------------------------
column: GK Handling
upper limit = 23.0
lower limit = -1.0
percentage of outliers= 10.181773520019652
'-----------------------------------------------
---------------------------------------------
column: GK Kicking
upper limit = 23.0
lower limit = -1.0
percentage of outliers= 10.206337509211497
'-----------------------------------------------
---------------------------------------------
column: GK Positioning
upper limit = 23.0
lower limit = -1.0
percentage of outliers= 10.187914517317614
'-----------------------------------------------
---------------------------------------------
column: GK Reflexes
upper limit = 23.0
lower limit = -1.0
percentage of outliers= 10.181773520019652
'-----------------------------------------------
---------------------------------------------
column: Total Stats
upper limit = 2226.5
lower limit = 1094.5
percentage of outliers= 4.925079832964874
'-----------------------------------------------
```

```
---------------------------------------------
column: Base Stats
upper limit = 461.5
lower limit = 265.5
percentage of outliers= 0.5404077622205846
'-----------------------------------------------
---------------------------------------------
column: PAC
upper limit = 94.5
lower limit = 42.5
percentage of outliers= 2.7511667894866125
'-----------------------------------------------
---------------------------------------------
column: SHO
upper limit = 93.5
lower limit = 17.5
percentage of outliers= 0.018422991893883568
'-----------------------------------------------
---------------------------------------------
column: PAS
upper limit = 81.5
lower limit = 37.5
percentage of outliers= 1.977401129943503
'-----------------------------------------------
---------------------------------------------
column: DRI
upper limit = 85.0
lower limit = 45.0
percentage of outliers= 4.630311962662736
'-----------------------------------------------
---------------------------------------------
column: DEF
upper limit = 104.5
lower limit = -3.5
percentage of outliers= 0.0
'-----------------------------------------------
---------------------------------------------
column: PHY
upper limit = 88.5
lower limit = 44.5
percentage of outliers= 1.9344141488577746
'-----------------------------------------------
---------------------------------------------
column: LB
upper limit = 0.0
```

```
lower limit = 0.0
percentage of outliers= 11.403831982313928
'-----------------------------------------------
---------------------------------------------
column: RW
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 7.90346352247605
'-----------------------------------------------
---------------------------------------------
column: CAM
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 11.895111766150823
'-----------------------------------------------
---------------------------------------------
column: CM
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 20.854826823876195
'-----------------------------------------------
---------------------------------------------
column: CF
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 2.0940800786047653
'-----------------------------------------------
---------------------------------------------
column: CB
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 20.903954802259886
'-----------------------------------------------
---------------------------------------------
column: RM
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 13.22156718251044
'-----------------------------------------------
---------------------------------------------
column: GK
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 10.12036354704004
'-----------------------------------------------
```

```
---------------------------------------------
column: RWB
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 1.989683124539425
'-----------------------------------------------
---------------------------------------------
column: LWB
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 2.0449521002210758
'-----------------------------------------------
---------------------------------------------
column: CDM
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 15.763940063866372
'-----------------------------------------------
---------------------------------------------
column: RB
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 11.323999017440432
'-----------------------------------------------
---------------------------------------------
column: ST
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 17.686072218128224
'-----------------------------------------------
---------------------------------------------
column: LM
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 13.553181036600344
'-----------------------------------------------
---------------------------------------------
column: LW
upper limit = 0.0
lower limit = 0.0
percentage of outliers= 7.897322525178089
'-----------------------------------------------
---------------------------------------------
column: W/F1
upper limit = 3.0
```

```
lower limit = 3.0
percentage of outliers= 38.350528125767624
'------------------------------------------------
----------------------------------------------
column: SM1
upper limit = 4.5
lower limit = 0.5
percentage of outliers= 0.31933185949398185
'------------------------------------------------
----------------------------------------------
column: IR1
upper limit = 1.0
lower limit = 1.0
percentage of outliers= 8.00786047654139
'------------------------------------------------
----------------------------------------------
column: weight
upper limit = 95.0
lower limit = 55.0
percentage of outliers= 0.39916482436747724
'------------------------------------------------
----------------------------------------------
column: height
upper limit = 201.0
lower limit = 161.0
percentage of outliers= 0.1658069270449521
'------------------------------------------------
----------------------------------------------
column: value
upper limit = 4850000.0
lower limit = -1950000.0
percentage of outliers= 12.331122574306068
'------------------------------------------------
----------------------------------------------
column: wage
upper limit = 54500.0
lower limit = -29500.0
percentage of outliers= 16.863178580201424
'------------------------------------------------
----------------------------------------------
column: Release clause
upper limit = 7806500.0
lower limit = -3677500.0
percentage of outliers= 14.247113731269959
'------------------------------------------------
```

```
---------------------------------------------
column: shortPass
upper limit = 87.0
lower limit = 39.0
percentage of outliers= 9.770326701056252
'---------------------------------------------
---------------------------------------------
column: dribbling
upper limit = 93.0
lower limit = 29.0
percentage of outliers= 10.973962171456645
'---------------------------------------------
```

In [ ]:

```

```

In [100…

```
# Checking for skweness.
skewness_dict = {}
for col in continous_vars[1:]:
    skewness = df_c[col].skew()
    skewness_dict[col] = skewness
    print(col, ':', skewness)
```

```
Age : 0.35508955319118213
↓OVA : 0.6718072997785406
POT : 0.3285112260616529
BOV : 0.6747728226730184
Attacking : -1.246947931910058
Crossing : -0.7834745163101987
Finishing : -0.41504855179016
Heading Accuracy : -0.9901924656809841
Volleys : -0.24333710100332742
Skill : -0.9609262893457415
Curve : -0.38325018178649933
FK Accuracy : 0.020689477908917415
Long Passing : -0.7437592521368035
Ball Control : -1.5782710043162147
Movement : -0.7611083585582729
Acceleration : -0.8094278485654471
Sprint Speed : -0.8253162083608858
Agility : -0.6751417630706197
Reactions : 0.14382249540107053
Balance : -0.6139936256430398
Power : -0.5816539675771503
Shot Power : -0.3507947088052558
Jumping : -0.46199397366497874
Stamina : -1.0296040661751589
Strength : -0.49802128492860587
Long Shots : -0.5613952225525299
Mentality : -1.1116739818416017
Aggression : -0.5601058899267962
Interceptions : -0.387449125108455
Positioning : -0.8811065398720058
Vision : -0.4770825953596053
Penalties : -0.38122133089075044
Composure : -0.5929113429070898
Defending : -0.4019128917077981
Marking : -0.449762285262942
Standing Tackle : -0.4535383868873854
Sliding Tackle : -0.36896915692622184
Goalkeeping : 2.662609040852694
GK Diving : 2.5711648236651716
GK Handling : 2.5643339572539103
GK Kicking : 2.5682697596129773
GK Positioning : 2.578372972695576
GK Reflexes : 2.580795920555938
Total Stats : -0.7705484001607902
Base Stats : 0.1653222364338735
```

```
PAC : -0.5844898039989985
SHO : -0.4820565758106385
PAS : -0.19386583085849707
DRI : -0.6030683219325025
DEF : -0.3155332341574588
PHY : -0.4939783185223727
LB : 2.4287416348776203
RW : 3.120944125201074
CAM : 2.3543250843450836
CM : 1.434898339246306
CF : 6.692032922966756
CB : 1.4312396289740525
RM : 2.171780120758171
GK : 2.6447978534639422
RWB : 6.8766468389227295
LWB : 6.777185174240479
CDM : 1.8791972646582906
RB : 2.4412294747645094
ST : 1.6939765492671308
LM : 2.129778825680169
LW : 3.122508674473604
W/F1 : 0.23671411890673053
SM1 : 0.12299869604458517
IR1 : 4.353019686418394
weight : 0.22979095390412024
height : -0.07043993624705805
value : 7.641739638734252
wage : 2.3923627534351435
Release clause : 6.596379305514824
shortPass : -1.4107305335581366
dribbling : -1.2848416443413244
```

In [ ]:

```
In [101…
# skewing the data
for col, skew in list(skewness_dict.items())[1:]:
    if skew >= 1:
        df_c[col] = np.log1p(df_c[col])
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_10820\3552965078.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a
-view-versus-a-copy
  df_c[col] = np.log1p(df_c[col])
```

In [ ]:

```

```

In [102…

```
# code to check if skewness made any changes
for col in continous_vars:
    skewed = df_c[col].skew()
    print(col, ':', skewed)
```

```
HITS : 31.339376836827498
Age : 0.35508955319118213
↓OVA : 0.6718072997785406
POT : 0.3285112260616529
BOV : 0.6747728226730184
Attacking : -1.246947931910058
Crossing : -0.7834745163101987
Finishing : -0.41504855179016
Heading Accuracy : -0.9901924656809841
Volleys : -0.24333710100332742
Skill : -0.9609262893457415
Curve : -0.38325018178649933
FK Accuracy : 0.020689477908917415
Long Passing : -0.7437592521368035
Ball Control : -1.5782710043162147
Movement : -0.7611083585582729
Acceleration : -0.8094278485654471
Sprint Speed : -0.8253162083608858
Agility : -0.6751417630706197
Reactions : 0.14382249540107053
Balance : -0.6139936256430398
Power : -0.5816539675771503
Shot Power : -0.3507947088052558
Jumping : -0.46199397366497874
Stamina : -1.0296040661751589
Strength : -0.49802128492860587
Long Shots : -0.5613952225525299
Mentality : -1.1116739818416017
Aggression : -0.5601058899267962
Interceptions : -0.387449125108455
Positioning : -0.8811065398720058
Vision : -0.4770825953596053
Penalties : -0.38122133089075044
Composure : -0.5929113429070898
Defending : -0.4019128917077981
Marking : -0.449762285262942
Standing Tackle : -0.4535383868873854
Sliding Tackle : -0.36896915692622184
Goalkeeping : 2.3295949465849004
GK Diving : 1.7271467474466724
GK Handling : 1.7064113083615577
GK Kicking : 1.6804220844207114
GK Positioning : 1.720257356871548
GK Reflexes : 1.7405177806556262
Total Stats : -0.7705484001607902
```

```
Base Stats : 0.1653222364338735
PAC : -0.5844898039989985
SHO : -0.4820565758106385
PAS : -0.19386583085849707
DRI : -0.6030683219325025
DEF : -0.3155332341574588
PHY : -0.4939783185223727
LB : 2.4287416348776207
RW : 3.1209441252010732
CAM : 2.3543250843450845
CM : 1.4348983392463064
CF : 6.692032922966754
CB : 1.4312396289740528
RM : 2.171780120758171
GK : 2.644797853463943
RWB : 6.876646838922731
LWB : 6.777185174240484
CDM : 1.8791972646582902
RB : 2.44122947476451
ST : 1.693976549267131
LM : 2.1297788256801695
LW : 3.1225086744736013
W/F1 : 0.23671411890673053
SM1 : 0.12299869604458517
IR1 : 3.726742559174215
weight : 0.22979095390412024
height : -0.07043993624705805
value : -4.42793811316795
wage : -0.16578118905633218
Release clause : -2.7443149037320786
shortPass : -1.4107305335581366
dribbling : -1.2848416443413244
```

In [ ]:

In [103…

```
# Checking the outliers and percentage to see if skewness made a change on it
for col in continous_vars:
    print("-------------------------------------------")
    print('Column : ',col)

    Ul,Ll = outlier_lims(df_c[col])
    print('Upper Limit = ',Ul)
    print('Lower Limit = ',Ll)

    total_outliers = len(df_c.loc[df_c[col]<Ll,col]) +len(df_c.loc[df_c[col]>Ul,col])
    percent = total_outliers / len(df_c.index) * 100

    print('Percentage of outliers = ',percent)
    print("-------------------------------------------")
```

```
-------------------------------------------
Column :  HITS
Upper Limit =  34.5
Lower Limit =  -17.5
Percentage of outliers =  13.135593220338984
-------------------------------------------
-------------------------------------------
Column :  Age
Upper Limit =  39.5
Lower Limit =  11.5
Percentage of outliers =  0.15352493244902973
-------------------------------------------
-------------------------------------------
Column :  ↓OVA
Upper Limit =  83.0
Lower Limit =  51.0
Percentage of outliers =  0.8720216163104888
-------------------------------------------
-------------------------------------------
Column :  POT
Upper Limit =  88.0
Lower Limit =  56.0
Percentage of outliers =  0.37460083517563253
-------------------------------------------
-------------------------------------------
Column :  BOV
Upper Limit =  84.0
Lower Limit =  52.0
Percentage of outliers =  0.7860476541390321
-------------------------------------------
-------------------------------------------
Column :  Attacking
Upper Limit =  402.5
Lower Limit =  134.5
Percentage of outliers =  10.249324490297223
-------------------------------------------
-------------------------------------------
Column :  Crossing
Upper Limit =  101.0
Lower Limit =  5.0
Percentage of outliers =  0.0
-------------------------------------------
-------------------------------------------
Column :  Finishing
Upper Limit =  109.5
```

```
Lower Limit =  -14.5
Percentage of outliers =  0.0
-------------------------------------------
-------------------------------------------
Column :  Heading Accuracy
Upper Limit =  93.5
Lower Limit =  17.5
Percentage of outliers =  8.21665438467207
-------------------------------------------
-------------------------------------------
Column :  Volleys
Upper Limit =  98.5
Lower Limit =  -9.5
Percentage of outliers =  0.0
-------------------------------------------
-------------------------------------------
Column :  Skill
Upper Limit =  435.0
Lower Limit =  115.0
Percentage of outliers =  9.217636944239745
-------------------------------------------
-------------------------------------------
Column :  Curve
Upper Limit =  102.0
Lower Limit =  -2.0
Percentage of outliers =  0.0
-------------------------------------------
-------------------------------------------
Column :  FK Accuracy
Upper Limit =  97.0
Lower Limit =  -7.0
Percentage of outliers =  0.0
-------------------------------------------
-------------------------------------------
Column :  Long Passing
Upper Limit =  93.5
Lower Limit =  17.5
Percentage of outliers =  1.2957504298698108
-------------------------------------------
-------------------------------------------
Column :  Ball Control
Upper Limit =  88.0
Lower Limit =  40.0
Percentage of outliers =  11.066077130926063
-------------------------------------------
```

```
-------------------------------------------
Column :  Movement
Upper Limit =  459.0
Lower Limit =  195.0
Percentage of outliers =  2.7143208056988453
-------------------------------------------
-------------------------------------------
Column :  Acceleration
Upper Limit =  100.5
Lower Limit =  32.5
Percentage of outliers =  3.641611397690985
-------------------------------------------
-------------------------------------------
Column :  Sprint Speed
Upper Limit =  100.5
Lower Limit =  32.5
Percentage of outliers =  3.2670105625153525
-------------------------------------------
-------------------------------------------
Column :  Agility
Upper Limit =  99.5
Lower Limit =  31.5
Percentage of outliers =  2.536231884057971
-------------------------------------------
-------------------------------------------
Column :  Reactions
Upper Limit =  85.5
Lower Limit =  41.5
Percentage of outliers =  0.7307786784573815
-------------------------------------------
-------------------------------------------
Column :  Balance
Upper Limit =  101.0
Lower Limit =  29.0
Percentage of outliers =  1.2220584622942767
-------------------------------------------
-------------------------------------------
Column :  Power
Upper Limit =  429.5
Lower Limit =  185.5
Percentage of outliers =  1.694915254237288
-------------------------------------------
-------------------------------------------
Column :  Shot Power
Upper Limit =  97.5
```

```
Lower Limit =  21.5
Percentage of outliers =  0.1105379513633014
-------------------------------------------
-------------------------------------------
Column :  Jumping
Upper Limit =  95.5
Lower Limit =  35.5
Percentage of outliers =  2.051093097519037
-------------------------------------------
-------------------------------------------
Column :  Stamina
Upper Limit =  96.5
Lower Limit =  36.5
Percentage of outliers =  8.640383198231392
-------------------------------------------
-------------------------------------------
Column :  Strength
Upper Limit =  96.5
Lower Limit =  36.5
Percentage of outliers =  1.9466961434536971
-------------------------------------------
-------------------------------------------
Column :  Long Shots
Upper Limit =  105.0
Lower Limit =  -7.0
Percentage of outliers =  0.0
-------------------------------------------
-------------------------------------------
Column :  Mentality
Upper Limit =  396.5
Lower Limit =  144.5
Percentage of outliers =  9.180790960451978
-------------------------------------------
-------------------------------------------
Column :  Aggression
Upper Limit =  104.5
Lower Limit =  12.5
Percentage of outliers =  0.07369196757553427
-------------------------------------------
-------------------------------------------
Column :  Interceptions
Upper Limit =  120.875
Lower Limit =  -28.125
Percentage of outliers =  0.0
-------------------------------------------
```

```
-------------------------------------------
Column :  Positioning
Upper Limit =  99.5
Lower Limit =  7.5
Percentage of outliers =  2.4318349299926307
-------------------------------------------
-------------------------------------------
Column :  Vision
Upper Limit =  92.0
Lower Limit =  20.0
Percentage of outliers =  0.6386637189879637
-------------------------------------------
-------------------------------------------
Column :  Penalties
Upper Limit =  92.5
Lower Limit =  8.5
Percentage of outliers =  0.04298698108572832
-------------------------------------------
-------------------------------------------
Column :  Composure
Upper Limit =  89.0
Lower Limit =  33.0
Percentage of outliers =  2.4318349299926307
-------------------------------------------
-------------------------------------------
Column :  Defending
Upper Limit =  354.5
Lower Limit =  -73.5
Percentage of outliers =  0.0
-------------------------------------------
-------------------------------------------
Column :  Marking
Upper Limit =  115.0
Lower Limit =  -21.0
Percentage of outliers =  0.0
-------------------------------------------
-------------------------------------------
Column :  Standing Tackle
Upper Limit =  124.0
Lower Limit =  -28.0
Percentage of outliers =  0.0
-------------------------------------------
-------------------------------------------
Column :  Sliding Tackle
Upper Limit =  121.0
```

```
Lower Limit =  -31.0
Percentage of outliers =  0.0
-------------------------------------------
-------------------------------------------
Column :  Goalkeeping
Upper Limit =  4.398130958389311
Lower Limit =  3.5880339019434153
Percentage of outliers =  11.336281012036356
-------------------------------------------
-------------------------------------------
Column :  GK Diving
Upper Limit =  3.474288636751196
Lower Limit =  1.4309861416872338
Percentage of outliers =  10.292311471382952
-------------------------------------------
-------------------------------------------
Column :  GK Handling
Upper Limit =  3.474288636751196
Lower Limit =  1.4309861416872338
Percentage of outliers =  10.316875460574797
-------------------------------------------
-------------------------------------------
Column :  GK Kicking
Upper Limit =  3.474288636751196
Lower Limit =  1.4309861416872338
Percentage of outliers =  10.304593465978874
-------------------------------------------
-------------------------------------------
Column :  GK Positioning
Upper Limit =  3.474288636751196
Lower Limit =  1.4309861416872338
Percentage of outliers =  10.304593465978874
-------------------------------------------
-------------------------------------------
Column :  GK Reflexes
Upper Limit =  3.474288636751196
Lower Limit =  1.4309861416872338
Percentage of outliers =  10.304593465978874
-------------------------------------------
-------------------------------------------
Column :  Total Stats
Upper Limit =  2226.5
Lower Limit =  1094.5
Percentage of outliers =  4.925079832964874
-------------------------------------------
```

```
-------------------------------------------
Column :  Base Stats
Upper Limit =  461.5
Lower Limit =  265.5
Percentage of outliers =  0.5404077622205846
-------------------------------------------
-------------------------------------------
Column :  PAC
Upper Limit =  94.5
Lower Limit =  42.5
Percentage of outliers =  2.7511667894866125
-------------------------------------------
-------------------------------------------
Column :  SHO
Upper Limit =  93.5
Lower Limit =  17.5
Percentage of outliers =  0.018422991893883568
-------------------------------------------
-------------------------------------------
Column :  PAS
Upper Limit =  81.5
Lower Limit =  37.5
Percentage of outliers =  1.977401129943503
-------------------------------------------
-------------------------------------------
Column :  DRI
Upper Limit =  85.0
Lower Limit =  45.0
Percentage of outliers =  4.630311962662736
-------------------------------------------
-------------------------------------------
Column :  DEF
Upper Limit =  104.5
Lower Limit =  -3.5
Percentage of outliers =  0.0
-------------------------------------------
-------------------------------------------
Column :  PHY
Upper Limit =  88.5
Lower Limit =  44.5
Percentage of outliers =  1.9344141488577746
-------------------------------------------
-------------------------------------------
Column :  LB
Upper Limit =  0.0
```

```
Lower Limit =  0.0
Percentage of outliers =  11.403831982313928
--------------------------------------------
--------------------------------------------
Column :  RW
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  7.90346352247605
--------------------------------------------
--------------------------------------------
Column :  CAM
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  11.895111766150823
--------------------------------------------
--------------------------------------------
Column :  CM
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  20.854826823876195
--------------------------------------------
--------------------------------------------
Column :  CF
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  2.0940800786047653
--------------------------------------------
--------------------------------------------
Column :  CB
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  20.903954802259886
--------------------------------------------
--------------------------------------------
Column :  RM
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  13.22156718251044
--------------------------------------------
--------------------------------------------
Column :  GK
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  10.12036354704004
--------------------------------------------
```

```
-------------------------------------------
Column :  RWB
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  1.989683124539425
-------------------------------------------
-------------------------------------------
Column :  LWB
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  2.0449521002210758
-------------------------------------------
-------------------------------------------
Column :  CDM
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  15.763940063866372
-------------------------------------------
-------------------------------------------
Column :  RB
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  11.323999017440432
-------------------------------------------
-------------------------------------------
Column :  ST
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  17.686072218128224
-------------------------------------------
-------------------------------------------
Column :  LM
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  13.553181036600344
-------------------------------------------
-------------------------------------------
Column :  LW
Upper Limit =  0.0
Lower Limit =  0.0
Percentage of outliers =  7.897322525178089
-------------------------------------------
-------------------------------------------
Column :  W/F1
Upper Limit =  3.0
```

```
Lower Limit =  3.0
Percentage of outliers =  38.350528125767624
-------------------------------------------
-------------------------------------------
Column :  SM1
Upper Limit =  4.5
Lower Limit =  0.5
Percentage of outliers =  0.31933185949398185
-------------------------------------------
-------------------------------------------
Column :  IR1
Upper Limit =  0.6931471805599453
Lower Limit =  0.6931471805599453
Percentage of outliers =  8.00786047654139
-------------------------------------------
-------------------------------------------
Column :  weight
Upper Limit =  95.0
Lower Limit =  55.0
Percentage of outliers =  0.39916482436747724
-------------------------------------------
-------------------------------------------
Column :  height
Upper Limit =  201.0
Lower Limit =  161.0
Percentage of outliers =  0.1658069270449521
-------------------------------------------
-------------------------------------------
Column :  value
Upper Limit =  16.66402038790939
Lower Limit =  11.289086328636067
Percentage of outliers =  5.490051584377302
-------------------------------------------
-------------------------------------------
Column :  wage
Upper Limit =  13.70612892869191
Lower Limit =  3.9385663781188276
Percentage of outliers =  3.0643576516826334
-------------------------------------------
-------------------------------------------
Column :  Release clause
Upper Limit =  17.64285234217507
Lower Limit =  10.777309595505251
Percentage of outliers =  8.431589290100712
-------------------------------------------
```

```
-------------------------------------------
Column :  shortPass
Upper Limit =  87.0
Lower Limit =  39.0
Percentage of outliers =  9.770326701056252
-------------------------------------------
-------------------------------------------
Column :  dribbling
Upper Limit =  93.0
Lower Limit =  29.0
Percentage of outliers =  10.973962171456645
-------------------------------------------
```

In [ ]:

```
In [104…
```

```python
# code to remove outlier
# Creating arrays of Boolean values indicating the outlier rows
for col in continous_vars:
    upper_array = np.where(df_c[col]>= UL)[0]
    lower_array = np.where(df_c[col]<=LL)[0]

# Dropping rows based on the indices in upper_array and lower_array
df_c.drop(index=upper_array, inplace=True)
df_c.drop(index=lower_array, inplace=True)

# Print the new shape of the DataFrame
print("New Shape: ", df_c.shape)
```

```
New Shape:  (14455, 88)
C:\Users\hp\AppData\Local\Temp\ipykernel_10820\2136527388.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a
-view-versus-a-copy
  df_c.drop(index=upper_array, inplace=True)
C:\Users\hp\AppData\Local\Temp\ipykernel_10820\2136527388.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a
-view-versus-a-copy
  df_c.drop(index=lower_array, inplace=True)
```

In [105…

```python
df_c.reset_index(drop = True,inplace = True)
```

In [106…
```
df_c
```

Out[106]:

	HITS	Age	↓OVA	POT	Preferred Foot	BOV	Best Position	Attacking	Crossing	Finishing	...	value	wage	Release clause	height_bins
0	562.0	35	92.0	92.0	Right	92	ST	437.0	84.0	95.0	...	17.958645	12.301387	18.144927	(180, 190]
1	207.0	29	91.0	91.0	Right	91	CAM	407.0	94.0	82.0	...	18.675323	12.821261	18.896915	(180, 190]
2	248.0	31	91.0	91.0	Right	91	ST	423.0	71.0	94.0	...	18.525041	12.388398	18.698312	(180, 190]
3	246.0	28	90.0	90.0	Left	90	RW	392.0	79.0	91.0	...	18.607160	12.429220	18.787405	(170, 180]
4	1600.0	21	90.0	95.0	Right	91	ST	408.0	78.0	91.0	...	19.038565	11.982935	19.129209	(170, 180]
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
14450	2.0	21	58.0	71.0	Right	60	CB	186.0	31.0	20.0	...	13.071072	13.217675	13.075274	(190, 200]
14451	1.0	19	58.0	73.0	Right	58	RB	208.0	55.0	31.0	...	13.071072	13.217675	13.110293	(170, 180]
14452	1.0	20	58.0	70.0	Right	58	LB	213.0	51.0	28.0	...	13.071072	13.764218	12.948012	(170, 180]
14453	1.0	25	58.0	62.0	Right	60	CM	232.0	35.0	48.0	...	12.524530	7.601402	12.584513	(170, 180]
14454	1.0	34	58.0	58.0	Left	58	CB	172.0	27.0	15.0	...	11.156265	13.217675	10.933125	(180, 190]

14455 rows × 88 columns

In [ ]:
```

```

In [107…
```
# 2. preprocess the cleaned data from task one above and transform it into a well behaved data.
```

```
['Preferred Foot', 'Best Position', 'A/W', 'D/W', 'playerName',
       'playerStatus', 'height_bins', 'weight_bins', 'wage_bins', 'value_bins',
       'Release clause_bins'],
```

Out[107]:
```
(['Preferred Foot',
  'Best Position',
  'A/W',
  'D/W',
  'playerName',
  'playerStatus',
  'height_bins',
  'weight_bins',
  'wage_bins',
  'value_bins',
  'Release clause_bins'],)
```

In [108…
```
# creating dummies for catgorical variables.
df_c = pd.get_dummies(df_c,columns=['Preferred Foot'])
df_c = pd.get_dummies(df_c,columns=['Best Position'])
df_c = pd.get_dummies(df_c,columns=['A/W'])
df_c = pd.get_dummies(df_c,columns=['D/W'])
df_c = pd.get_dummies(df_c,columns=['playerStatus'])
df_c = pd.get_dummies(df_c,columns=['height_bins'])
df_c = pd.get_dummies(df_c,columns=['weight_bins'])
df_c = pd.get_dummies(df_c,columns=['wage_bins'])
df_c = pd.get_dummies(df_c,columns=['value_bins'])
df_c = pd.get_dummies(df_c,columns=['Release clause_bins'])
```

In [109…
```
df_c
```

Out[109]:

	HITS	Age	↓OVA	POT	BOV	Attacking	Crossing	Finishing	Heading Accuracy	Volleys	...	wage_bins_[950000, 960000)	value_bins_[0, 50000000)	value_bins_[5 10
0	562.0	35	92.0	92.0	92	437.0	84.0	95.0	90.0	86.0	...	0	0	
1	207.0	29	91.0	91.0	91	407.0	94.0	82.0	55.0	82.0	...	0	0	
2	248.0	31	91.0	91.0	91	423.0	71.0	94.0	85.0	89.0	...	0	0	
3	246.0	28	90.0	90.0	90	392.0	79.0	91.0	59.0	79.0	...	0	0	
4	1600.0	21	90.0	95.0	91	408.0	78.0	91.0	73.0	83.0	...	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
14450	2.0	21	58.0	71.0	60	186.0	31.0	20.0	58.0	26.0	...	0	1	
14451	1.0	19	58.0	73.0	58	208.0	55.0	31.0	47.0	28.0	...	0	1	
14452	1.0	20	58.0	70.0	58	213.0	51.0	28.0	48.0	28.0	...	1	1	
14453	1.0	25	58.0	62.0	60	232.0	35.0	48.0	51.0	35.0	...	0	1	
14454	1.0	34	58.0	58.0	58	172.0	27.0	15.0	58.0	28.0	...	0	1	

14455 rows × 162 columns

In [ ]:

In [110…

```
# RESACLING,NORMILIZATION AND STANDARDIZATION
df_c.describe().T
```

Out[110]:

	count	mean	std	min	25%	50%	75%	max
HITS	14455.0	27.811553	135.504134	1.0	2.0	5.0	15.0	8400.0
Age	14455.0	25.718782	4.391551	16.0	22.0	25.0	29.0	53.0
↓OVA	14455.0	67.491249	5.585695	58.0	63.0	67.0	71.0	92.0
POT	14455.0	71.938845	5.854137	58.0	68.0	72.0	76.0	95.0
BOV	14455.0	68.538637	5.437407	58.0	64.0	68.0	72.0	92.0
...	...	...	...	...	...	...	...	...
Release clause_bins_[0, 50000000)	14455.0	0.989900	0.099995	0.0	1.0	1.0	1.0	1.0
Release clause_bins_[50000000, 100000000)	14455.0	0.008440	0.091484	0.0	0.0	0.0	0.0	1.0
Release clause_bins_[100000000, 150000000)	14455.0	0.001522	0.038984	0.0	0.0	0.0	0.0	1.0
Release clause_bins_[150000000, 200000000)	14455.0	0.000069	0.008317	0.0	0.0	0.0	0.0	1.0
Release clause_bins_[200000000, 250000000)	14455.0	0.000069	0.008317	0.0	0.0	0.0	0.0	1.0

161 rows × 8 columns

In [111…
```
# conclusion about my data using df_c.describe()
# 1. the difference in the scale is very large, therefore my Data need resacling
# 2. the data need restandardization
# 3. The data is not normally distributed therefore i need to normilzed the data.
```

In [ ]:
```

```

In [112…
```
# RESCALING
scaler = MinMaxScaler()
for col in continous_vars[1:]:
    df_c[col] = scaler.fit_transform(df_c[[col]])
```

In [113…
```
df_c.describe().T
```

Out[113]:

	count	mean	std	min	25%	50%	75%	max
HITS	14455.0	27.811553	135.504134	1.0	2.000000	5.000000	15.000000	8400.0
Age	14455.0	0.262670	0.118691	0.0	0.162162	0.243243	0.351351	1.0
↓OVA	14455.0	0.279154	0.164285	0.0	0.147059	0.264706	0.382353	1.0
POT	14455.0	0.376726	0.158220	0.0	0.270270	0.378378	0.486486	1.0
BOV	14455.0	0.309960	0.159924	0.0	0.176471	0.294118	0.411765	1.0
...	...	...	...	...	...	...	...	...
Release clause_bins_[0, 50000000)	14455.0	0.989900	0.099995	0.0	1.000000	1.000000	1.000000	1.0
Release clause_bins_[50000000, 100000000)	14455.0	0.008440	0.091484	0.0	0.000000	0.000000	0.000000	1.0
Release clause_bins_[100000000, 150000000)	14455.0	0.001522	0.038984	0.0	0.000000	0.000000	0.000000	1.0
Release clause_bins_[150000000, 200000000)	14455.0	0.000069	0.008317	0.0	0.000000	0.000000	0.000000	1.0
Release clause_bins_[200000000, 250000000)	14455.0	0.000069	0.008317	0.0	0.000000	0.000000	0.000000	1.0

161 rows × 8 columns

In [ ]:

```
```

In [114…

```
# checking for all the column to be sure my data doesnt need standardization
# according to the result, there is no standard error in my data.
# meaning i won"t be restandardizing my data.
for cols in continous_vars:
    print(f'column name : {df_c[cols]}\nstd: {df_c[cols].std()}')
```

```
column name : 0         562.0
1         207.0
2         248.0
3         246.0
4        1600.0
          ...
14450       2.0
14451       1.0
14452       1.0
14453       1.0
14454       1.0
Name: HITS, Length: 14455, dtype: float64
std: 135.50413397576867
column name : 0        0.513514
1        0.351351
2        0.405405
3        0.324324
4        0.135135
           ...
14450    0.135135
14451    0.081081
14452    0.108108
14453    0.243243
14454    0.486486
Name: Age, Length: 14455, dtype: float64
std: 0.11869057842684848
column name : 0        1.000000
1        0.970588
2        0.970588
3        0.941176
4        0.941176
           ...
14450    0.000000
14451    0.000000
14452    0.000000
14453    0.000000
14454    0.000000
Name: ↓OVA, Length: 14455, dtype: float64
std: 0.16428514838106287
column name : 0        0.918919
1        0.891892
2        0.891892
3        0.864865
4        1.000000
           ...
```

```
14450    0.351351
14451    0.405405
14452    0.324324
14453    0.108108
14454    0.000000
Name: POT, Length: 14455, dtype: float64
std: 0.1582199286310056
column name : 0        1.000000
1        0.970588
2        0.970588
3        0.941176
4        0.970588
           ...
14450    0.058824
14451    0.000000
14452    0.000000
14453    0.058824
14454    0.000000
Name: BOV, Length: 14455, dtype: float64
std: 0.15992372659388807
column name : 0        1.000000
1        0.916667
2        0.961111
3        0.875000
4        0.919444
           ...
14450    0.302778
14451    0.363889
14452    0.377778
14453    0.430556
14454    0.263889
Name: Attacking, Length: 14455, dtype: float64
std: 0.11663514638571146
column name : 0        0.879518
1        1.000000
2        0.722892
3        0.819277
4        0.807229
           ...
14450    0.240964
14451    0.530120
14452    0.481928
14453    0.289157
14454    0.192771
Name: Crossing, Length: 14455, dtype: float64
```

```
std: 0.1578305075462958
column name : 0        1.000000
1        0.852273
2        0.988636
3        0.954545
4        0.954545
           ...
14450    0.147727
14451    0.272727
14452    0.238636
14453    0.465909
14454    0.090909
Name: Finishing, Length: 14455, dtype: float64
std: 0.18070708618739295
column name : 0        0.963415
1        0.536585
2        0.902439
3        0.585366
4        0.756098
           ...
14450    0.573171
14451    0.439024
14452    0.451220
14453    0.487805
14454    0.573171
Name: Heading Accuracy, Length: 14455, dtype: float64
std: 0.13976856733880189
column name : 0        0.952381
1        0.904762
2        0.988095
3        0.869048
4        0.916667
           ...
14450    0.238095
14451    0.261905
14452    0.261905
14453    0.345238
14454    0.261905
Name: Volleys, Length: 14455, dtype: float64
std: 0.17353671971409781
column name : 0        0.917431
1        1.000000
2        0.896024
3        0.892966
4        0.856269
```

```
          ...
14450    0.165138
14451    0.186544
14452    0.305810
14453    0.403670
14454    0.119266
Name: Skill, Length: 14455, dtype: float64
std: 0.1519933540006874
column name : 0        0.860759
1        0.911392
2        0.835443
3        0.886076
4        0.835443
          ...
14450    0.126582
14451    0.189873
14452    0.227848
14453    0.316456
14454    0.101266
Name: Curve, Length: 14455, dtype: float64
std: 0.18388733742350627
column name : 0        0.795181
1        0.879518
2        0.903614
3        0.710843
4        0.638554
          ...
14450    0.144578
14451    0.228916
14452    0.289157
14453    0.240964
14454    0.132530
Name: FK Accuracy, Length: 14455, dtype: float64
std: 0.1784700508066747
column name : 0        0.780822
1        1.000000
2        0.684932
3        0.753425
4        0.684932
          ...
14450    0.301370
14451    0.041096
14452    0.410959
14453    0.520548
14454    0.027397
```

```
Name: Long Passing, Length: 14455, dtype: float64
std: 0.15444058764530535
column name : 0        0.972222
1        0.972222
2        0.916667
3        0.930556
4        0.944444
           ...
14450    0.291667
14451    0.291667
14452    0.347222
14453    0.527778
14454    0.361111
Name: Ball Control, Length: 14455, dtype: float64
std: 0.11421787061753903
column name : 0        0.888514
1        0.777027
2        0.807432
3        0.986486
4        0.979730
           ...
14450    0.304054
14451    0.550676
14452    0.510135
14453    0.364865
14454    0.141892
Name: Movement, Length: 14455, dtype: float64
std: 0.14351281757561826
column name : 0        0.863014
1        0.726027
2        0.726027
3        0.958904
4        0.986301
           ...
14450    0.410959
14451    0.712329
14452    0.575342
14453    0.273973
14454    0.205479
Name: Acceleration, Length: 14455, dtype: float64
std: 0.16020914005317202
column name : 0        0.929577
1        0.718310
2        0.746479
3        0.943662
```

```
4        1.000000
           ...
14450    0.549296
14451    0.661972
14452    0.521127
14453    0.338028
14454    0.267606
Name: Sprint Speed, Length: 14455, dtype: float64
std: 0.1606754596188178
column name : 0        0.884058
1        0.753623
2        0.739130
3        0.942029
4        0.956522
           ...
14450    0.304348
14451    0.521739
14452    0.463768
14453    0.565217
14454    0.072464
Name: Agility, Length: 14455, dtype: float64
std: 0.17233844038159654
column name : 0        1.000000
1        0.929825
2        0.964912
3        0.947368
4        0.947368
           ...
14450    0.228070
14451    0.175439
14452    0.350877
14453    0.350877
14454    0.298246
Name: Reactions, Length: 14455, dtype: float64
std: 0.13484022812886523
column name : 0        0.643836
1        0.712329
2        0.794521
3        0.917808
4        0.794521
           ...
14450    0.246575
14451    0.671233
14452    0.698630
14453    0.493151
```

```
14454    0.232877
Name: Balance, Length: 14455, dtype: float64
std: 0.16466744135023872
column name : 0        1.000000
1        0.861538
2        0.907692
3        0.803846
4        0.846154
           ...
14450    0.288462
14451    0.153846
14452    0.173077
14453    0.415385
14454    0.161538
Name: Power, Length: 14455, dtype: float64
std: 0.14241636487928422
column name : 0        1.000000
1        0.960526
2        0.934211
3        0.815789
4        0.894737
           ...
14450    0.460526
14451    0.105263
14452    0.250000
14453    0.381579
14454    0.210526
Name: Shot Power, Length: 14455, dtype: float64
std: 0.16658739792217384
column name : 0        1.000000
1        0.536232
2        0.840580
3        0.623188
4        0.739130
           ...
14450    0.405797
14451    0.449275
14452    0.449275
14453    0.753623
14454    0.449275
Name: Jumping, Length: 14455, dtype: float64
std: 0.1690663626140938
column name : 0        0.828947
1        0.894737
2        0.723684
```

```
3        0.842105
4        0.855263
           ...
14450    0.447368
14451    0.578947
14452    0.486842
14453    0.552632
14454    0.328947
Name: Stamina, Length: 14455, dtype: float64
std: 0.14005323209936157
column name : 0        0.739726
1        0.684932
2        0.849315
3        0.698630
4        0.712329
           ...
14450    0.684932
14451    0.397260
14452    0.301370
14453    0.383562
14454    0.616438
Name: Strength, Length: 14455, dtype: float64
std: 0.1647735330303147
column name : 0        1.000000
1        0.976471
2        0.905882
3        0.894118
4        0.835294
           ...
14450    0.176471
14451    0.176471
14452    0.270588
14453    0.517647
14454    0.141176
Name: Long Shots, Length: 14455, dtype: float64
std: 0.17847396866532364
column name : 0        0.788820
1        0.959627
2        0.906832
3        0.860248
4        0.751553
           ...
14450    0.326087
14451    0.397516
14452    0.409938
```

```
14453    0.475155
14454    0.354037
Name: Mentality, Length: 14455, dtype: float64
std: 0.11888430241413164
column name : 0        0.582278
1        0.746835
2        0.810127
3        0.582278
4        0.569620
           ...
14450    0.506329
14451    0.493671
14452    0.392405
14453    0.202532
14454    0.594937
Name: Aggression, Length: 14455, dtype: float64
std: 0.17466145244137823
column name : 0        0.234568
1        0.691358
2        0.481481
3        0.555556
4        0.345679
           ...
14450    0.567901
14451    0.567901
14452    0.543210
14453    0.530864
14454    0.580247
Name: Interceptions, Length: 14455, dtype: float64
std: 0.23001644281169673
column name : 0        1.000000
1        0.922222
2        0.988889
3        0.955556
4        0.955556
           ...
14450    0.255556
14451    0.366667
14452    0.511111
14453    0.533333
14454    0.211111
Name: Positioning, Length: 14455, dtype: float64
std: 0.15363465191811332
column name : 0        0.851852
1        1.000000
```

```
2        0.814815
3        0.876543
4        0.827160
           ...
14450    0.246914
14451    0.283951
14452    0.358025
14453    0.530864
14454    0.209877
Name: Vision, Length: 14455, dtype: float64
std: 0.149777010158334
column name : 0        0.898734
1        0.898734
2        0.949367
3        0.886076
4        0.721519
           ...
14450    0.215190
14451    0.354430
14452    0.291139
14453    0.556962
14454    0.316456
Name: Penalties, Length: 14455, dtype: float64
std: 0.15563997805557472
column name : 0        1.000000
1        0.929825
2        0.877193
3        0.912281
4        0.807018
           ...
14450    0.105263
14451    0.122807
14452    0.140351
14453    0.087719
14454    0.017544
Name: Composure, Length: 14455, dtype: float64
std: 0.16190779403243072
column name : 0        0.206751
1        0.637131
2        0.257384
3        0.367089
4        0.274262
           ...
14450    0.590717
14451    0.544304
```

```
14452    0.565401
14453    0.552743
14454    0.582278
Name: Defending, Length: 14455, dtype: float64
std: 0.2266076147330055
column name : 0        0.214286
1        0.690476
2        0.297619
3        0.333333
4        0.285714
           ...
14450    0.571429
14451    0.452381
14452    0.488095
14453    0.535714
14454    0.571429
Name: Marking, Length: 14455, dtype: float64
std: 0.20754946366225327
column name : 0        0.265060
1        0.662651
2        0.385542
3        0.397590
4        0.289157
           ...
14450    0.602410
14451    0.554217
14452    0.626506
14453    0.566265
14454    0.602410
Name: Standing Tackle, Length: 14455, dtype: float64
std: 0.2268641623154882
column name : 0        0.1750
1        0.5375
2        0.1125
3        0.3875
4        0.2750
          ...
14450    0.5875
14451    0.6250
14452    0.5750
14453    0.5500
14454    0.5625
Name: Sliding Tackle, Length: 14455, dtype: float64
std: 0.23633213389151692
column name : 0        0.455046
```

```
1        0.445703
2        0.420831
3        0.472817
4        0.369344
           ...
14450    0.404732
14451    0.415570
14452    0.335855
14453    0.425991
14454    0.415570
Name: Goalkeeping, Length: 14455, dtype: float64
std: 0.04266907975342465
column name : 0        0.290296
1        0.495446
2        0.495446
3        0.476345
4        0.455925
           ...
14450    0.356339
14451    0.290296
14452    0.290296
14453    0.325156
14454    0.325156
Name: GK Diving, Length: 14455, dtype: float64
std: 0.08461360781334493
column name : 0        0.410301
1        0.455925
2        0.250774
3        0.476345
4        0.205151
           ...
14450    0.410301
14451    0.476345
14452    0.384548
14453    0.433991
14454    0.455925
Name: GK Handling, Length: 14455, dtype: float64
std: 0.08355116661438322
column name : 0        0.489013
1        0.202487
2        0.428356
3        0.351712
4        0.286526
           ...
14450    0.470159
```

```
14451    0.450005
14452    0.247518
14453    0.379555
14454    0.379555
Name: GK Kicking, Length: 14455, dtype: float64
std: 0.0834974365882396
column name : 0        0.477962
1        0.385854
2        0.326260
3        0.411694
4        0.411694
           ...
14450    0.326260
14451    0.291281
14452    0.357549
14453    0.457472
14454    0.326260
Name: GK Positioning, Length: 14455, dtype: float64
std: 0.08409471865131535
column name : 0        0.407590
1        0.452913
2        0.382008
3        0.473197
4        0.249118
           ...
14450    0.249118
14451    0.353985
14452    0.203795
14453    0.353985
14454    0.407590
Name: GK Reflexes, Length: 14455, dtype: float64
std: 0.08387958965456765
column name : 0        0.917031
1        0.989520
2        0.894323
3        0.908297
4        0.852402
           ...
14450    0.110917
14451    0.181659
14452    0.210480
14453    0.301310
14454    0.022707
Name: Total Stats, Length: 14455, dtype: float64
std: 0.15240034791241386
```

```
column name : 0        0.861345
1        0.949580
2        0.831933
3        0.886555
4        0.869748
           ...
14450    0.134454
14451    0.193277
14452    0.189076
14453    0.235294
14454    0.000000
Name: Base Stats, Length: 14455, dtype: float64
std: 0.14733403115299504
column name : 0        0.901408
1        0.718310
2        0.746479
3        0.957746
4        1.000000
           ...
14450    0.492958
14451    0.690141
14452    0.549296
14453    0.309859
14454    0.239437
Name: PAC, Length: 14455, dtype: float64
std: 0.15756248261291417
column name : 0        1.000000
1        0.909091
2        0.974026
3        0.909091
4        0.909091
           ...
14450    0.155844
14451    0.168831
14452    0.207792
14453    0.428571
14454    0.077922
Name: SHO, Length: 14455, dtype: float64
std: 0.17555448184077843
column name : 0        0.820896
1        1.000000
2        0.776119
3        0.820896
4        0.776119
           ...
```

```
14450    0.194030
14451    0.223881
14452    0.358209
14453    0.388060
14454    0.089552
Name: PAS, Length: 14455, dtype: float64
std: 0.13714474615137548
column name : 0        0.949153
1        0.932203
2        0.881356
3        0.966102
4        0.983051
           ...
14450    0.135593
14451    0.305085
14452    0.338983
14453    0.457627
14454    0.169492
Name: DRI, Length: 14455, dtype: float64
std: 0.15167894538781157
column name : 0        0.253333
1        0.640000
2        0.360000
3        0.386667
4        0.306667
           ...
14450    0.560000
14451    0.493333
14452    0.520000
14453    0.520000
14454    0.560000
Name: DEF, Length: 14455, dtype: float64
std: 0.21718678337574646
column name : 0        0.774194
1        0.790323
2        0.854839
3        0.741935
4        0.758065
           ...
14450    0.580645
14451    0.451613
14452    0.338710
14453    0.370968
14454    0.532258
Name: PHY, Length: 14455, dtype: float64
```

```
std: 0.14637228765597948
column name : 0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
        ...
14450    0.0
14451    0.0
14452    1.0
14453    0.0
14454    0.0
Name: LB, Length: 14455, dtype: float64
std: 0.3338516097217857
column name : 0        0.0
1        0.0
2        0.0
3        1.0
4        1.0
        ...
14450    0.0
14451    0.0
14452    0.0
14453    0.0
14454    0.0
Name: RW, Length: 14455, dtype: float64
std: 0.2846042480319542
column name : 0        0.0
1        1.0
2        0.0
3        0.0
4        0.0
        ...
14450    0.0
14451    0.0
14452    0.0
14453    0.0
14454    0.0
Name: CAM, Length: 14455, dtype: float64
std: 0.3405176095166029
column name : 0        0.0
1        1.0
2        0.0
3        0.0
4        0.0
```

```
        ...
14450    0.0
14451    0.0
14452    0.0
14453    1.0
14454    0.0
Name: CM, Length: 14455, dtype: float64
std: 0.42393022303270095
column name : 0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
        ...
14450    0.0
14451    0.0
14452    0.0
14453    0.0
14454    0.0
Name: CF, Length: 14455, dtype: float64
std: 0.15155728843914232
column name : 0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
        ...
14450    1.0
14451    0.0
14452    0.0
14453    0.0
14454    1.0
Name: CB, Length: 14455, dtype: float64
std: 0.41643537212618786
column name : 0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
        ...
14450    0.0
14451    1.0
14452    0.0
14453    0.0
14454    0.0
```

```
Name: RM, Length: 14455, dtype: float64
std: 0.35604656732914686
column name : 0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
        ...
14450    0.0
14451    0.0
14452    0.0
14453    0.0
14454    0.0
Name: GK, Length: 14455, dtype: float64
std: 0.014405274241642717
column name : 0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
        ...
14450    0.0
14451    0.0
14452    0.0
14453    0.0
14454    0.0
Name: RWB, Length: 14455, dtype: float64
std: 0.1478087017109823
column name : 0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
        ...
14450    0.0
14451    0.0
14452    0.0
14453    0.0
14454    0.0
Name: LWB, Length: 14455, dtype: float64
std: 0.15002621310529504
column name : 0        0.0
1        0.0
2        0.0
3        0.0
```

```
4        0.0
        ...
14450    0.0
14451    0.0
14452    0.0
14453    1.0
14454    0.0
Name: CDM, Length: 14455, dtype: float64
std: 0.3817091296991286
column name : 0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
        ...
14450    1.0
14451    1.0
14452    0.0
14453    0.0
14454    0.0
Name: RB, Length: 14455, dtype: float64
std: 0.3324579112891162
column name : 0        1.0
1        0.0
2        1.0
3        0.0
4        1.0
        ...
14450    0.0
14451    0.0
14452    0.0
14453    0.0
14454    0.0
Name: ST, Length: 14455, dtype: float64
std: 0.39928558387773394
column name : 0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
        ...
14450    0.0
14451    0.0
14452    0.0
14453    0.0
```

```
14454    0.0
Name: LM, Length: 14455, dtype: float64
std: 0.3596919201248566
column name : 0        1.0
1        0.0
2        0.0
3        0.0
4        1.0
        ...
14450    0.0
14451    0.0
14452    0.0
14453    0.0
14454    0.0
Name: LW, Length: 14455, dtype: float64
std: 0.28450428613673345
column name : 0        0.75
1        1.00
2        0.75
3        0.50
4        0.75
         ...
14450    0.75
14451    0.50
14452    0.75
14453    0.25
14454    0.25
Name: W/F1, Length: 14455, dtype: float64
std: 0.16249397399770185
column name : 0        1.00
1        0.75
2        0.75
3        0.75
4        1.00
         ...
14450    0.25
14451    0.25
14452    0.25
14453    0.25
14454    0.25
Name: SM1, Length: 14455, dtype: float64
std: 0.16086628077755613
column name : 0        1.000000
1        0.834044
2        0.834044
```

```
3        0.630930
4        0.630930
           ...
14450    0.000000
14451    0.000000
14452    0.000000
14453    0.000000
14454    0.000000
Name: IR1, Length: 14455, dtype: float64
std: 0.1263308597094696
column name : 0        0.550000
1        0.333333
2        0.500000
3        0.350000
4        0.383333
           ...
14450    0.516667
14451    0.300000
14452    0.250000
14453    0.466667
14454    0.633333
Name: weight, Length: 14455, dtype: float64
std: 0.11158136533599863
column name : 0        0.683794
1        0.565217
2        0.624506
3        0.446640
4        0.505929
           ...
14450    0.782609
14451    0.486166
14452    0.446640
14453    0.446640
14454    0.664032
Name: height, Length: 14455, dtype: float64
std: 0.1298059551451476
column name : 0        0.943277
1        0.980921
2        0.973027
3        0.977340
4        1.000000
           ...
14450    0.686558
14451    0.686558
14452    0.686558
```

```
14453    0.657851
14454    0.585982
Name: value, Length: 14455, dtype: float64
std: 0.1042924175931606
column name : 0        0.893722
1        0.931492
2        0.900044
3        0.903010
4        0.870586
           ...
14450    0.960292
14451    0.960292
14452    1.000000
14453    0.552258
14454    0.960292
Name: wage, Length: 14455, dtype: float64
std: 0.15806145900678706
column name : 0        0.948546
1        0.987857
2        0.977474
3        0.982132
4        1.000000
           ...
14450    0.683524
14451    0.685355
14452    0.676871
14453    0.657869
14454    0.571541
Name: Release clause, Length: 14455, dtype: float64
std: 0.20655160747778087
column name : 0        0.828571
1        1.000000
2        0.857143
3        0.857143
4        0.842857
           ...
14450    0.385714
14451    0.328571
14452    0.485714
14453    0.557143
14454    0.285714
Name: shortPass, Length: 14455, dtype: float64
std: 0.11454200236855959
column name : 0        0.935484
1        0.935484
```

```
2        0.887097
3        0.967742
4        1.000000
           ...
14450    0.129032
14451    0.354839
14452    0.354839
14453    0.483871
14454    0.177419
Name: dribbling, Length: 14455, dtype: float64
std: 0.17555902804455037
```

In [ ]:

In [115…
```
# concluion after resacling and runing df_.describe
# 1. the data is rescaled
# 2. there is no standard error in the data
# 3. The data is normally distributed.
```

In [ ]:

In [116…
```
# dropping the column playerName cause the datatype is object.
df_c = df_c.drop (['playerName'],axis = 1)
```

In [ ]:

In [117…
```
# 3. select input features for an outcome feature of HITS.
# outcome column is HITS
y = df_c.HITS.values
x = df_c.values[:,1:]
```

In [118…
```
y
```

Out[118]:
```
array([562., 207., 248., ...,   1.,   1.,   1.])
```

In [119…
```
x
```

```
Out[119]: array([[0.51351351, 1.        , 0.91891892, ..., 0.        , 0.        ,
                 0.        ],
                [0.35135135, 0.97058824, 0.89189189, ..., 0.        , 1.        ,
                 0.        ],
                [0.40540541, 0.97058824, 0.89189189, ..., 1.        , 0.        ,
                 0.        ],
                ...,
                [0.10810811, 0.        , 0.32432432, ..., 0.        , 0.        ,
                 0.        ],
                [0.24324324, 0.        , 0.10810811, ..., 0.        , 0.        ,
                 0.        ],
                [0.48648649, 0.        , 0.        , ..., 0.        , 0.        ,
                 0.        ]])
```

```
In [120…
from pandas import read_csv
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression
 #load data

y = df_c.HITS.values
X = df_c.values [:,1:]
#feature extraction
model = LogisticRegression(solver='lbfgs')
rfe = RFE(model, n_features_to_select=82)
fit = rfe.fit(X, y)
print("Num Features:%d" % fit.n_features_)
print("selected Features:%s" % fit.support_)
print("Feature Ranking: %s" % fit.ranking_)
```

```
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
Num Features:82
selected Features:[ True  True  True  True False False False False False False False  True
  True False False  True  True False  True False False False  True  True
  True  True False  True  True False False  True  True False  True False
  True False  True  True  True  True  True  True  True False False False
  True False False  True  True  True  True  True  True  True False False
  True  True  True  True  True  True  True  True  True False False  True
 False  True False  True  True  True  True  True  True False  True False
  True  True False  True  True  True False  True  True  True  True  True
  True  True  True  True  True False False False False False False False
 False False False False False False False False  True  True  True  True
 False False False False False False  True  True  True False False  True
  True  True False False False False False False False  True False False
 False False False False False False False  True  True False False  True
  True False False False]
Feature Ranking: [ 1  1  1  1 47 12 26  6  3 40  9  1  1 19 25  1  1 17  1 23 28 22  1  1
  1  1 45  1  1  5 31  1  1 21  1  4  1 15  1  1  1  1  1  1  1 14 32 33
  1 27 10  1  1  1  1  1  1  1 55  7  1  1  1  1  1  1  1  1  1 13 18  1
 20  1 16  1  1  1  1  1  1 30  1 54  1  1 24  1  1  1  8  1  1  1  1  1
  1  1  1  1  1 71 70 74 75 79 69 68 67 65 66 76 77 64 63 46  1  1  1  1
 50 61 78 62 57 11  1  1  1  2 52  1  1  1 37 48 51 53 56 72 73  1 38 41
 42 39 29 43 35 34 36  1  1 49 58  1  1 44 60 59]
```

```
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

In [ ]:

```
```

In [122…

```
model.fit(X,y)

score = model.score(X,y)
```

```
C:\Users\hp\Documents\data analysis workspace\lib\site-packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarni
ng: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

In [123…
```
score
```

Out[123]:
```
0.19121411276374956
```

In [ ]:
```
```