

[Open in Colab](#)

```
In [1]: #Data Analysis Libraries
import numpy as np
import pandas as pd

#Visualisation Libraries
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

#Ignore Warnings
import warnings
warnings.filterwarnings('ignore')

#Google Colab Import Files
from google.colab import files
import io
```

```
In [2]: uploaded = files.upload()
```

Choose files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

Saving nomis_2024_03_04_132215.csv to nomis_2024_03_04_132215.csv
 Saving NS-SEC_2021.csv to NS-SEC_2021.csv
 Saving table-1-2020-21.csv to table-1-2020-21.csv

```
In [3]: GradData = pd.read_csv('table-1-2020-21.csv')
AreaQuali = pd.read_csv('nomis_2024_03_04_132215.csv')
SERank = pd.read_csv('NS-SEC_2021.csv')
```

```
In [4]: print(GradData.columns)
print(AreaQuali.columns)
print(SERank.columns)
```

```
Index(['UKPRN', 'Provider name', 'Country of provider', 'Domicile',
       'Level of qualification obtained', 'Mode of former study',
       'Interim study', 'Sex', 'Activity', 'Academic year', 'Number'],
      dtype='object')
Index(['2022 local authorities: county', 'No qualifications',
       'Level 1 and entry level qualifications', 'Level 2 qualifications',
       'Apprenticeship', 'Level 3 qualifications',
       'Level 4 qualifications or above', 'Other qualifications', 'Unnamed: 8',
       'Unnamed: 9'],
      dtype='object')
Index(['ONSConstID', 'ConstituencyName', 'RegNationID', 'RegNationName',
       'NatComparator', 'variables', 'groups', 'Con_num', 'Con_pc', 'RN_pc',
       'Nat_pc', 'ranking_total', 'rank'],
      dtype='object')
```

```
In [5]: AreaQuali = AreaQuali.drop(['Unnamed: 8'], axis = 1)
AreaQuali = AreaQuali.drop(['Unnamed: 9'], axis = 1)
```

```
In [6]: GradData.astype('string')
AreaQuali.astype('string')
SERank.astype('string')
```

Out[6]:

	ONSConstID	ConstituencyName	RegNationID	RegNationName	NatComparator	variable
0	E14000530	Aldershot	E12000008	South East	England & Wales	High manager administr and profess
1	E14000530	Aldershot	E12000008	South East	England & Wales	Low manager administr a professi
2	E14000530	Aldershot	E12000008	South East	England & Wales	Intermediate occupati
3	E14000530	Aldershot	E12000008	South East	England & Wales	Small employe and o' accou work
4	E14000530	Aldershot	E12000008	South East	England & Wales	Low supervis and techni occupati
...
5152	W07000080	Cardiff South and Penarth	W92000004	Wales	England & Wales	Low supervis and techni occupati
5153	W07000080	Cardiff South and Penarth	W92000004	Wales	England & Wales	Semi-routi occupati
5154	W07000080	Cardiff South and Penarth	W92000004	Wales	England & Wales	Route occupati
5155	W07000080	Cardiff South and Penarth	W92000004	Wales	England & Wales	Never work / long-te unemploy
5156	W07000080	Cardiff South and Penarth	W92000004	Wales	England & Wales	Full-ti studie

5157 rows × 13 columns

In [7]:

```
AreaQuali['No qualifications'].astype('int')
AreaQuali['Level 1 and entry level qualifications'].astype('int')
AreaQuali['Level 2 qualifications'].astype('int')
AreaQuali['Apprenticeship'].astype('int')
AreaQuali['Level 3 qualifications'].astype('int')
AreaQuali['Level 4 qualifications or above'].astype('int')
AreaQuali['Other qualifications'].astype('int')
```

```
Out[7]: 0      789
        1      3199
        2      602
        3     1034
        4     1906
        ...
       169    1412
       170    586
       171    735
       172    584
       173   1597
Name: Other qualifications, Length: 174, dtype: int64
```

```
In [8]: SERank['rank'].astype('int')
SERank['ranking_total'].astype('float')
```

```
Out[8]: 0      0.337408
        1      0.337408
        2      0.230380
        3      0.230380
        4      0.303069
        ...
       5152   0.278226
       5153   0.278226
       5154   0.278226
       5155   0.106746
       5156   0.092428
Name: ranking_total, Length: 5157, dtype: float64
```

```
In [9]: GradData.describe(include = "all")
```

	UKPRN	Provider name	Country of provider	Domicile	Level of qualification obtained	Mode of former study	Interim study	Sex
count	1.048561e+06	1048561	1048561	1048561	1048561	1048561	1048561	1048561
unique	NaN	340	5	4	3	3	2	4
top	NaN	The Open University	All	All	All	All	Include significant interim study	All
freq	NaN	6656	522880	409663	405997	394115	524862	329004
mean	1.000964e+07	NaN	NaN	NaN	NaN	NaN	NaN	NaN
std	1.331511e+04	NaN	NaN	NaN	NaN	NaN	NaN	NaN
min	1.000006e+07	NaN	NaN	NaN	NaN	NaN	NaN	NaN
25%	1.000386e+07	NaN	NaN	NaN	NaN	NaN	NaN	NaN
50%	1.000714e+07	NaN	NaN	NaN	NaN	NaN	NaN	NaN
75%	1.000782e+07	NaN	NaN	NaN	NaN	NaN	NaN	NaN
max	1.008659e+07	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
In [10]: GradData.head()
```

Out[10]:

	UKPRN	Provider name	Country of provider	Domicile	Level of qualification obtained	Mode of former study	Interim study	Sex	Ac
0	10008071	AA School of Architecture	England	Non-UK	All	Full-time	Exclude significant interim study	Female	Full employ
1	10008071	AA School of Architecture	England	Non-UK	All	Full-time	Exclude significant interim study	Female	Part employ
2	10008071	AA School of Architecture	England	Non-UK	All	Full-time	Exclude significant interim study	Female	Unk pattern employ
3	10008071	AA School of Architecture	England	Non-UK	All	Full-time	Exclude significant interim study	Female	Volunt unpaid
4	10008071	AA School of Architecture	England	Non-UK	All	Full-time	Exclude significant interim study	Female	Employer further

In [11]: AreaQuali.describe(include = "all")

Out[11]:

	2022 local authorities: county	No qualifications	Level 1 and entry level qualifications	Level 2 qualifications	Apprenticeship	Level 3 qualifications
count	174	174.000000	174.000000	174.000000	174.000000	174.000000
unique	174	NaN	NaN	NaN	NaN	NaN
top	Darlington	NaN	NaN	NaN	NaN	NaN
freq	1	NaN	NaN	NaN	NaN	NaN
mean	NaN	13713.195402	13821.396552	20142.356322	7447.798851	29858.931034
std	NaN	10604.969794	12443.469521	18300.281003	6565.655899	26931.198757
min	NaN	61.000000	77.000000	170.000000	38.000000	255.000000
25%	NaN	6665.750000	6540.000000	9446.000000	3592.250000	14637.500000
50%	NaN	11037.500000	10131.000000	13420.000000	5103.000000	19999.000000
75%	NaN	17239.750000	15180.750000	22032.500000	8482.000000	34047.500000
max	NaN	60591.000000	77586.000000	113671.000000	36082.000000	150678.000000

In [12]: AreaQuali.head()

Out[12]:

	2022 local authorities: county	No qualifications	Level 1 and entry level qualifications	Level 2 qualifications	Apprenticeship	Level 3 qualifications	qua
0	Darlington	4306	4190	6925	2845	11297	
1	County Durham	19076	19757	33256	12329	51751	
2	Hartlepool	3450	3108	5649	2250	9155	
3	Middlesbrough	6289	4950	7551	3215	11811	
4	Northumberland	9939	12398	20706	8349	32548	

In [13]:

SERank.describe(include = "all")

Out[13]:

	ONSConstID	ConstituencyName	RegNationID	RegNationName	NatComparator	varia
count	5157		5157	5157	5157	5157
unique	573		573	10	10	1
top	E14000530	Aldershot	E12000008	South East	England & Wales	Hi manag administr and prof
freq	9		9	756	756	5157
mean	NaN		NaN	NaN	NaN	NaN
std	NaN		NaN	NaN	NaN	NaN
min	NaN		NaN	NaN	NaN	NaN
25%	NaN		NaN	NaN	NaN	NaN
50%	NaN		NaN	NaN	NaN	NaN
75%	NaN		NaN	NaN	NaN	NaN
max	NaN		NaN	NaN	NaN	NaN

In [14]:

SERank.head()

	ONSConstID	ConstituencyName	RegNationID	RegNationName	NatComparator	variables
0	E14000530	Aldershot	E12000008	South East	England & Wales	Higher managerial, administrative and professi...
1	E14000530	Aldershot	E12000008	South East	England & Wales	Lower managerial, administrative and professio...
2	E14000530	Aldershot	E12000008	South East	England & Wales	Intermediate occupations
3	E14000530	Aldershot	E12000008	South East	England & Wales	Small employers and own account workers
4	E14000530	Aldershot	E12000008	South East	England & Wales	Lower supervisory and technical occupations

◀ ▶

```
In [15]: GradData = GradData.drop(['UKPRN'], axis = 1)
GradData = GradData.drop(['Country of provider'], axis = 1)
GradData = GradData.drop(['Domicile'], axis = 1)
GradData = GradData.drop(['Interim study'], axis = 1)
GradData = GradData.drop(['Sex'], axis = 1)
GradData = GradData.drop(['Academic year'], axis = 1)
```

```
In [16]: GradData.rename(columns={'Number': 'Total'}, inplace=True)
```

```
In [17]: SERank = SERank.drop(['ONSConstID'], axis = 1)
SERank = SERank.drop(['RegNationID'], axis = 1)
SERank = SERank.drop(['NatComparator'], axis = 1)
SERank = SERank.drop(['variables'], axis = 1)
SERank = SERank.drop(['Con_num'], axis = 1)
SERank = SERank.drop(['Con_pc'], axis = 1)
SERank = SERank.drop(['RN_pc'], axis = 1)
SERank = SERank.drop(['Nat_pc'], axis = 1)
```

```
In [18]: AreaQuali.set_index('2022 local authorities: county', inplace=True)
AreaQuali = AreaQuali.stack().to_frame('Total').reset_index()
AreaQuali.rename(columns={'level_1':'Qualification Type'},inplace=True)
```

```
In [19]: GradData['Activity'].unique()
```

```
Out[19]: array(['Full-time employment', 'Part-time employment',
       'Unknown pattern of employment', 'Voluntary or unpaid work',
       'Employment\x00and further study', 'Full-time further study',
       'Part-time further study', 'Unknown pattern of further study',
       'Other including travel, caring for someone or retired',
       'Unemployed', 'Total with known outcomes', 'Non-respondents',
       'Total'], dtype=object)
```

```
In [20]: maskA = GradData['Activity'] == 'Total with known outcomes'
maskB = GradData['Activity'] == 'Total'
maskC = GradData['Activity'] == 'Non-respondents'

GradData = GradData[~maskA]
GradData = GradData[~maskB]
GradData = GradData[~maskC]
```

```
In [21]: combine = [GradData]

for dataset in combine:
    dataset['Activity'] = dataset['Activity'].replace(['Full-time employment', 'Par
    dataset['Activity'] = dataset['Activity'].replace(['Full-time further study', 'P
    dataset['Activity'] = dataset['Activity'].replace(['Employment\xa0and further s
    dataset['Activity'] = dataset['Activity'].replace(['Other including travel, car
```

```
In [22]: AreaQuali['Qualification Type'].unique()
```

```
Out[22]: array(['No qualifications', 'Level 1 and entry level qualifications',
               'Level 2 qualifications', 'Apprenticeship',
               'Level 3 qualifications', 'Level 4 qualifications or above',
               'Other qualifications'], dtype=object)
```

```
In [23]: combine = [AreaQuali]

for dataset in combine:
    dataset['Qualification Type'] = dataset['Qualification Type'].replace(['Level 1
```

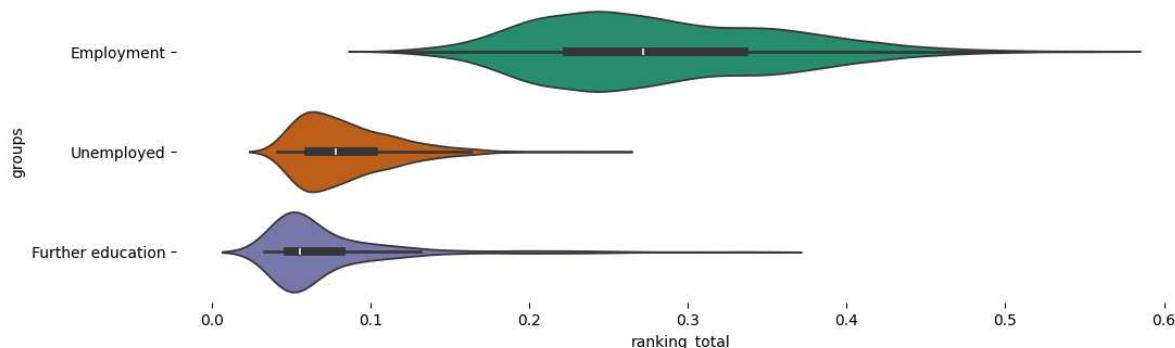
```
In [24]: SERank['groups'].unique()
```

```
Out[24]: array(['Managerial, administrative and professional occupations',
               'Intermediate occupations', 'Routine and manual occupations',
               'Never worked / long-term unemployed', 'Full-time students'],
               dtype=object)
```

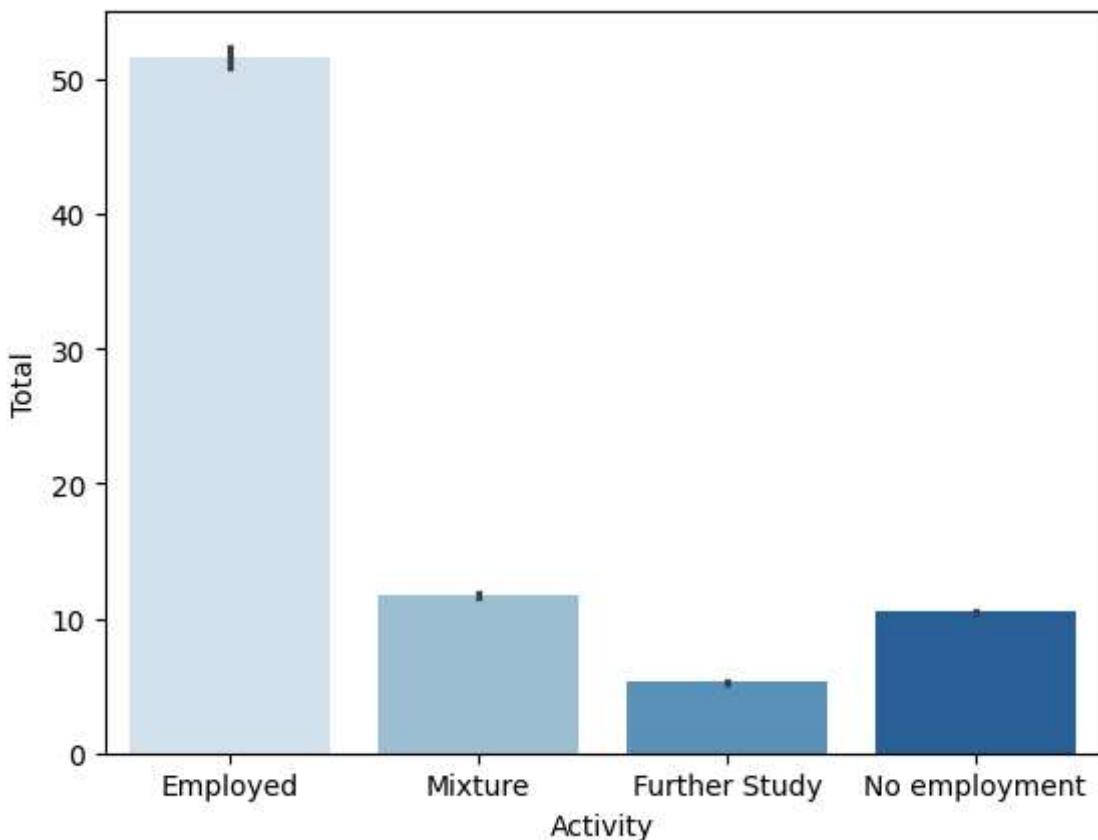
```
In [25]: combine = [SERank]

for dataset in combine:
    dataset['groups'] = dataset['groups'].replace(['Managerial, administrative and
    dataset['groups'] = dataset['groups'].replace('Never worked / long-term unempl
    dataset['groups'] = dataset['groups'].replace('Full-time students', 'Further ed
```

```
In [26]: figsize = (12, 1.2 * len(SERank['groups'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(SERank, x='ranking_total', y='groups', inner='box', palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```



```
In [27]: sns.barplot(x = "Activity", y = "Total", data = GradData, palette = "Blues")
plt.show()
```



```
In [28]: top5rn = SERank.sort_values(by=['rank']).head(5)
bottom5rn = SERank.sort_values(by=['rank']).tail(5)

print(top5rn)
print(bottom5rn)
```

	ConstituencyName	RegNationName	groups
3041	Nottingham South	East Midlands	Further education
172	Battersea	London	Employment
171	Battersea	London	Employment
304	Birmingham, Hodge Hill	West Midlands	Unemployed
1678	Great Grimsby	Yorkshire and The Humber	Employment

	ranking_total	rank	ConstituencyName	RegNationName	groups
3041	0.345145	1	Nottingham South	East Midlands	Further education
172	0.556306	1	Battersea	London	Employment
171	0.556306	1	Battersea	London	Employment
304	0.247469	1	Birmingham, Hodge Hill	West Midlands	Unemployed
1678	0.458726	1	Great Grimsby	Yorkshire and The Humber	Employment
3300	0.114850	573	Richmond Park	London	Employment
224	0.032813	573	Berwick-upon-Tweed	North East	Further education
2833	0.041182	573	North East Hampshire	South East	Unemployed
297	0.136152	573	Birmingham, Hodge Hill	West Midlands	Employment
785	0.129088	573	Cambridge	East of England	Employment

	ranking_total	rank
3300	0.114850	573
224	0.032813	573
2833	0.041182	573
297	0.136152	573
785	0.129088	573

```
In [29]: top5rt = SERank.sort_values(by=['ranking_total']).tail(5)
bottom5rt = SERank.sort_values(by=['ranking_total']).head(5)
```

```
print(top5rt)
print(bottom5rt)
```

	ConstituencyName	RegNationName	groups	ranking_total	rank
4590	Wimbledon	London	Employment	0.542545	3
3295	Richmond Park	London	Employment	0.551775	2
3294	Richmond Park	London	Employment	0.551775	2
172	Battersea	London	Employment	0.556306	1
171	Battersea	London	Employment	0.556306	1
	ConstituencyName	RegNationName	groups	ranking_total	\
224	Berwick-upon-Tweed	North East	Further education	0.032813	
2870	North Norfolk	East of England	Further education	0.033315	
1016	Clacton	East of England	Further education	0.035385	
2420	Louth and Horncastle	East Midlands	Further education	0.036881	
2969	North West Norfolk	East of England	Further education	0.037842	
	rank				
224	573				
2870	572				
1016	571				
2420	570				
2969	569				

In [30]: `AreaQuali['2022 local authorities: county'].unique()`

```
Out[30]: array(['Darlington', 'County Durham', 'Hartlepool', 'Middlesbrough',
   'Northumberland', 'Redcar and Cleveland', 'Stockton-on-Tees',
   'Gateshead', 'Newcastle upon Tyne', 'North Tyneside',
   'South Tyneside', 'Sunderland', 'Blackburn with Darwen',
   'Blackpool', 'Cheshire East', 'Cheshire West and Chester',
   'Halton', 'Warrington', 'Cumbria', 'Bolton', 'Bury', 'Manchester',
   'Oldham', 'Rochdale', 'Salford', 'Stockport', 'Tameside',
   'Trafford', 'Wigan', 'Lancashire', 'Knowsley', 'Liverpool',
   'Sefton', 'St. Helens', 'Wirral', 'East Riding of Yorkshire',
   'Kingston upon Hull, City of', 'North East Lincolnshire',
   'North Lincolnshire', 'York', 'North Yorkshire', 'Barnsley',
   'Doncaster', 'Rotherham', 'Sheffield', 'Bradford', 'Calderdale',
   'Kirklees', 'Leeds', 'Wakefield', 'Derby', 'Leicester',
   'Nottingham', 'Rutland', 'Derbyshire', 'Leicestershire',
   'Lincolnshire', 'Nottinghamshire', 'North Northamptonshire',
   'West Northamptonshire', 'Herefordshire, County of', 'Shropshire',
   'Stoke-on-Trent', 'Telford and Wrekin', 'Staffordshire',
   'Warwickshire', 'Birmingham', 'Coventry', 'Dudley', 'Sandwell',
   'Solihull', 'Walsall', 'Wolverhampton', 'Worcestershire',
   'Bedford', 'Central Bedfordshire', 'Luton', 'Peterborough',
   'Southend-on-Sea', 'Thurrock', 'Cambridgeshire', 'Essex',
   'Hertfordshire', 'Norfolk', 'Suffolk', 'Camden', 'City of London',
   'Hackney', 'Hammersmith and Fulham', 'Haringey', 'Islington',
   'Kensington and Chelsea', 'Lambeth', 'Lewisham', 'Newham',
   'Southwark', 'Tower Hamlets', 'Wandsworth', 'Westminster',
   'Barking and Dagenham', 'Barnet', 'Bexley', 'Brent', 'Bromley',
   'Croydon', 'Ealing', 'Enfield', 'Greenwich', 'Harrow', 'Havering',
   'Hillingdon', 'Hounslow', 'Kingston upon Thames', 'Merton',
   'Redbridge', 'Richmond upon Thames', 'Sutton', 'Waltham Forest',
   'Bracknell Forest', 'Brighton and Hove', 'Isle of Wight', 'Medway',
   'Milton Keynes', 'Portsmouth', 'Reading', 'Slough', 'Southampton',
   'West Berkshire', 'Windsor and Maidenhead', 'Wokingham',
   'Buckinghamshire', 'East Sussex', 'Hampshire', 'Kent',
   'Oxfordshire', 'Surrey', 'West Sussex',
   'Bath and North East Somerset', 'Bristol, City of', 'Cornwall',
   'Isles of Scilly', 'North Somerset', 'Plymouth',
   'Bournemouth, Christchurch and Poole', 'South Gloucestershire',
   'Swindon', 'Torbay', 'Wiltshire', 'Devon', 'Dorset',
   'Gloucestershire', 'Somerset', 'Isle of Anglesey', 'Gwynedd',
   'Conwy', 'Denbighshire', 'Flintshire', 'Wrexham', 'Powys',
   'Ceredigion', 'Pembrokeshire', 'Carmarthenshire', 'Swansea',
   'Neath Port Talbot', 'Bridgend', 'Vale of Glamorgan', 'Cardiff',
   'Rhondda Cynon Taff', 'Merthyr Tydfil', 'Caerphilly',
   'Blaenau Gwent', 'Torfaen', 'Monmouthshire', 'Newport'],
  dtype=object)
```

```
In [31]: Nott = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Nottinghamshire']
Mert = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Merton']
Brum = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Birmingham']
NELinc = AreaQuali[AreaQuali['2022 local authorities: county'] == 'North East Lincolnshire']

Top5rn = pd.concat([Nott, Mert, Brum, NELinc], ignore_index=True)
Top5rn
```

Out[31]:

	2022 local authorities: county	Qualification Type	Total
0	Nottinghamshire	No qualifications	32562
1	Nottinghamshire	Other qualifications	36725
2	Nottinghamshire	Other qualifications	54099
3	Nottinghamshire	Apprenticeship	21070
4	Nottinghamshire	Other qualifications	82962
5	Nottinghamshire	Level 4 qualifications or above	144056
6	Nottinghamshire	Other qualifications	6788
7	Merton	No qualifications	8956
8	Merton	Other qualifications	6526
9	Merton	Other qualifications	7942
10	Merton	Apprenticeship	3309
11	Merton	Other qualifications	13453
12	Merton	Level 4 qualifications or above	66101
13	Merton	Other qualifications	2925
14	Birmingham	No qualifications	51860
15	Birmingham	Other qualifications	39489
16	Birmingham	Other qualifications	50221
17	Birmingham	Apprenticeship	15585
18	Birmingham	Other qualifications	73407
19	Birmingham	Level 4 qualifications or above	180826
20	Birmingham	Other qualifications	10082
21	North East Lincolnshire	No qualifications	8121
22	North East Lincolnshire	Other qualifications	8120
23	North East Lincolnshire	Other qualifications	11793
24	North East Lincolnshire	Apprenticeship	4097
25	North East Lincolnshire	Other qualifications	14612
26	North East Lincolnshire	Level 4 qualifications or above	18869
27	North East Lincolnshire	Other qualifications	1497

In [32]:

```
Rich = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Richmond upon Than  
Nland = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Northumberland']  
NNshire = AreaQuali[AreaQuali['2022 local authorities: county'] == 'North Northampt  
Brum = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Birmingham']  
Camb = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Cambridgeshire']
```

```
Bottom5rn = pd.concat([Rich, Nland, NNshire, Brum, Camb], ignore_index=True)  
Bottom5rn
```

Out[32]:

	2022 local authorities: county	Qualification Type	Total
0	Richmond upon Thames	No qualifications	3751
1	Richmond upon Thames	Other qualifications	3523
2	Richmond upon Thames	Other qualifications	5500
3	Richmond upon Thames	Apprenticeship	1804
4	Richmond upon Thames	Other qualifications	10829
5	Richmond upon Thames	Level 4 qualifications or above	68908
6	Richmond upon Thames	Other qualifications	1409
7	Northumberland	No qualifications	9939
8	Northumberland	Other qualifications	12398
9	Northumberland	Other qualifications	20706
10	Northumberland	Apprenticeship	8349
11	Northumberland	Other qualifications	32548
12	Northumberland	Level 4 qualifications or above	52321
13	Northumberland	Other qualifications	1906
14	North Northamptonshire	No qualifications	19048
15	North Northamptonshire	Other qualifications	20148
16	North Northamptonshire	Other qualifications	27306
17	North Northamptonshire	Apprenticeship	10441
18	North Northamptonshire	Other qualifications	35858
19	North Northamptonshire	Level 4 qualifications or above	57404
20	North Northamptonshire	Other qualifications	4482
21	Birmingham	No qualifications	51860
22	Birmingham	Other qualifications	39489
23	Birmingham	Other qualifications	50221
24	Birmingham	Apprenticeship	15585
25	Birmingham	Other qualifications	73407
26	Birmingham	Level 4 qualifications or above	180826
27	Birmingham	Other qualifications	10082
28	Cambridgeshire	No qualifications	24125
29	Cambridgeshire	Other qualifications	27012
30	Cambridgeshire	Other qualifications	37974
31	Cambridgeshire	Apprenticeship	14593
32	Cambridgeshire	Other qualifications	57412
33	Cambridgeshire	Level 4 qualifications or above	158494
34	Cambridgeshire	Other qualifications	6200

```
In [33]: Mert = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Merton']
Rich = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Richmond upon Than
Wand = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Wandsworth']

Top5rt = pd.concat([Mert, Rich, Wand], ignore_index=True)
Top5rt
```

Out[33]:

	2022 local authorities: county	Qualification Type	Total
0	Merton	No qualifications	8956
1	Merton	Other qualifications	6526
2	Merton	Other qualifications	7942
3	Merton	Apprenticeship	3309
4	Merton	Other qualifications	13453
5	Merton	Level 4 qualifications or above	66101
6	Merton	Other qualifications	2925
7	Richmond upon Thames	No qualifications	3751
8	Richmond upon Thames	Other qualifications	3523
9	Richmond upon Thames	Other qualifications	5500
10	Richmond upon Thames	Apprenticeship	1804
11	Richmond upon Thames	Other qualifications	10829
12	Richmond upon Thames	Level 4 qualifications or above	68908
13	Richmond upon Thames	Other qualifications	1409
14	Wandsworth	No qualifications	8538
15	Wandsworth	Other qualifications	5933
16	Wandsworth	Other qualifications	8431
17	Wandsworth	Apprenticeship	2985
18	Wandsworth	Other qualifications	17872
19	Wandsworth	Level 4 qualifications or above	139034
20	Wandsworth	Other qualifications	3431

```
In [34]: Nland = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Northumberland']
Norf = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Norfolk']
Ess = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Essex']
Linc = AreaQuali[AreaQuali['2022 local authorities: county'] == 'Lincolnshire']

Bottom5rt = pd.concat([Nland, Norf, Ess, Linc], ignore_index=True)
Bottom5rt
```

Out[34]:

	2022 local authorities: county	Qualification Type	Total
0	Northumberland	No qualifications	9939
1	Northumberland	Other qualifications	12398
2	Northumberland	Other qualifications	20706
3	Northumberland	Apprenticeship	8349
4	Northumberland	Other qualifications	32548
5	Northumberland	Level 4 qualifications or above	52321
6	Northumberland	Other qualifications	1906
7	Norfolk	No qualifications	39678
8	Norfolk	Other qualifications	44773
9	Norfolk	Other qualifications	63952
10	Norfolk	Apprenticeship	23836
11	Norfolk	Other qualifications	87385
12	Norfolk	Level 4 qualifications or above	135372
13	Norfolk	Other qualifications	8544
14	Essex	No qualifications	60591
15	Essex	Other qualifications	77586
16	Essex	Other qualifications	113671
17	Essex	Apprenticeship	36082
18	Essex	Other qualifications	150678
19	Essex	Level 4 qualifications or above	245544
20	Essex	Other qualifications	13546
21	Lincolnshire	No qualifications	35024
22	Lincolnshire	Other qualifications	36001
23	Lincolnshire	Other qualifications	51953
24	Lincolnshire	Apprenticeship	20577
25	Lincolnshire	Other qualifications	76817
26	Lincolnshire	Level 4 qualifications or above	110851
27	Lincolnshire	Other qualifications	8224

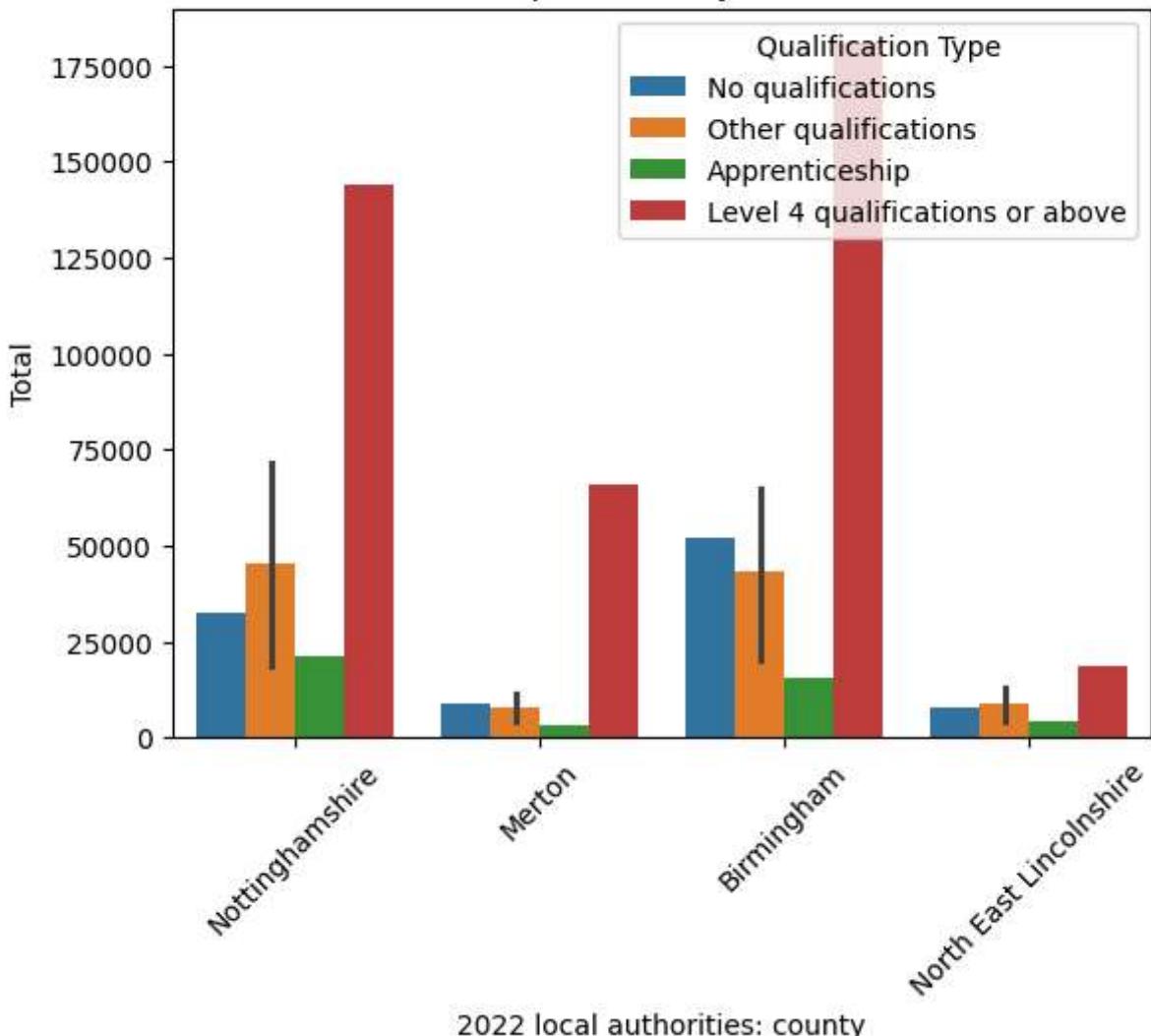
In [35]:

```
sns.barplot(data=Top5rn, x='2022 local authorities: county', y='Total', hue='Qualif
plt.xticks(rotation=45)
```

Out[35]:

```
([0, 1, 2, 3],
 [Text(0, 0, 'Nottinghamshire'),
  Text(1, 0, 'Merton'),
  Text(2, 0, 'Birmingham'),
  Text(3, 0, 'North East Lincolnshire')])
```

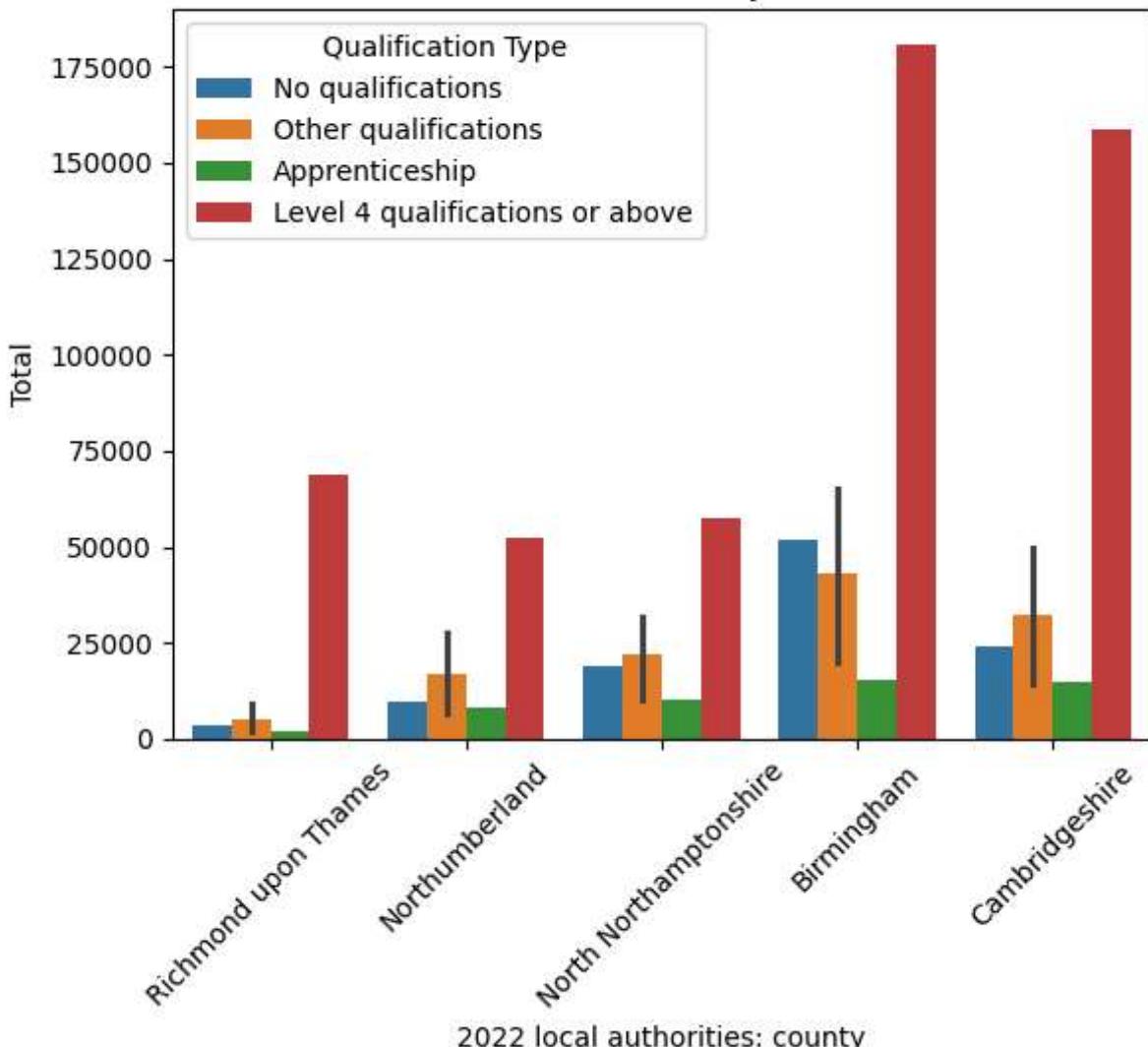
Top 5 Areas by Rank



```
In [36]: sns.barplot(data=Bottom5rn, x='2022 local authorities: county', y='Total', hue='Qualification Type')
plt.xticks(rotation=45)
```

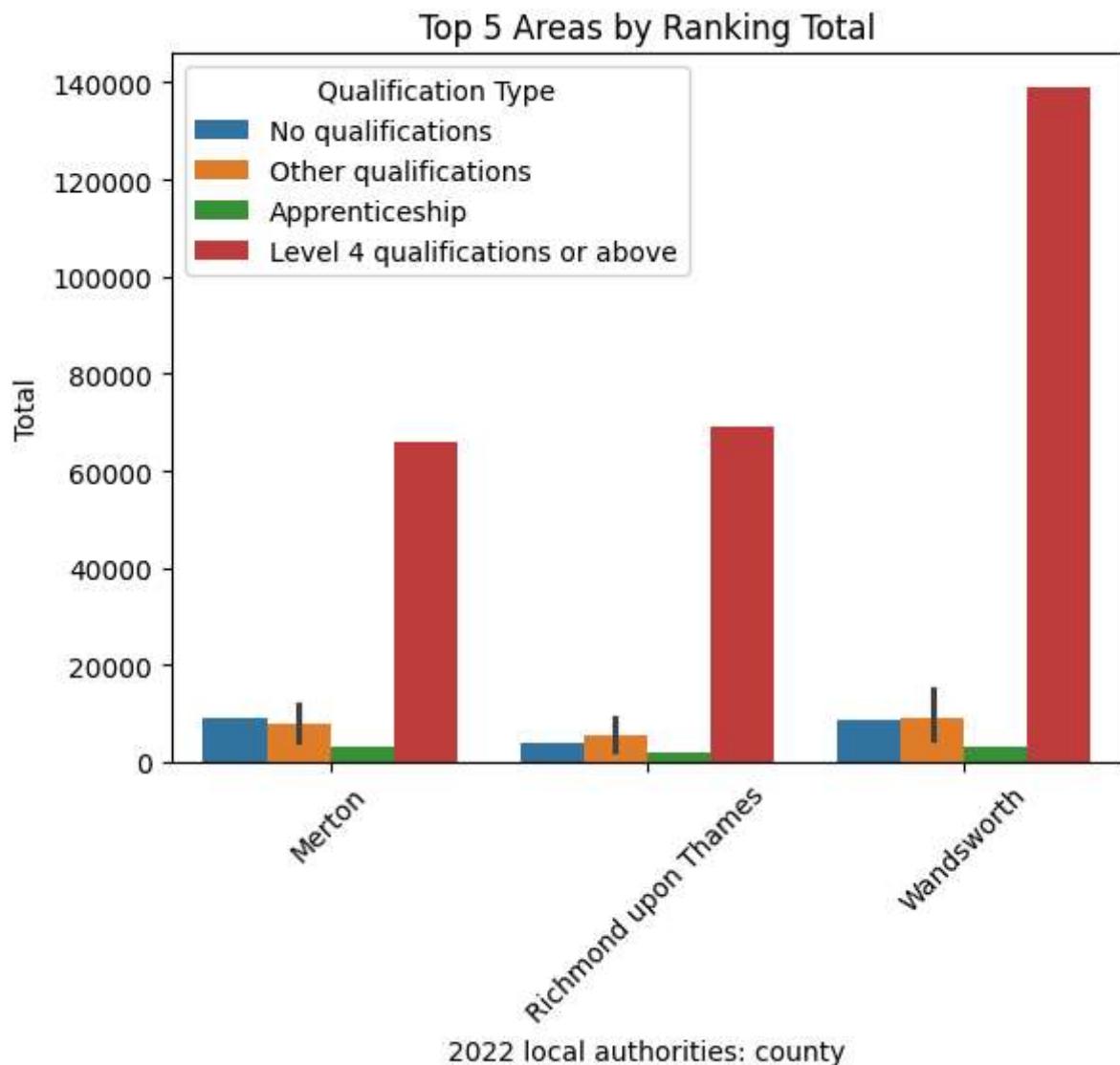
```
Out[36]: ([0, 1, 2, 3, 4],
[Text(0, 0, 'Richmond upon Thames'),
 Text(1, 0, 'Northumberland'),
 Text(2, 0, 'North Northamptonshire'),
 Text(3, 0, 'Birmingham'),
 Text(4, 0, 'Cambridgeshire')])
```

Bottom 5 Areas by Rank



```
In [37]: sns.barplot(data=Top5rt, x='2022 local authorities: county', y='Total', hue='Qualification Type')
plt.xticks(rotation=45)
```

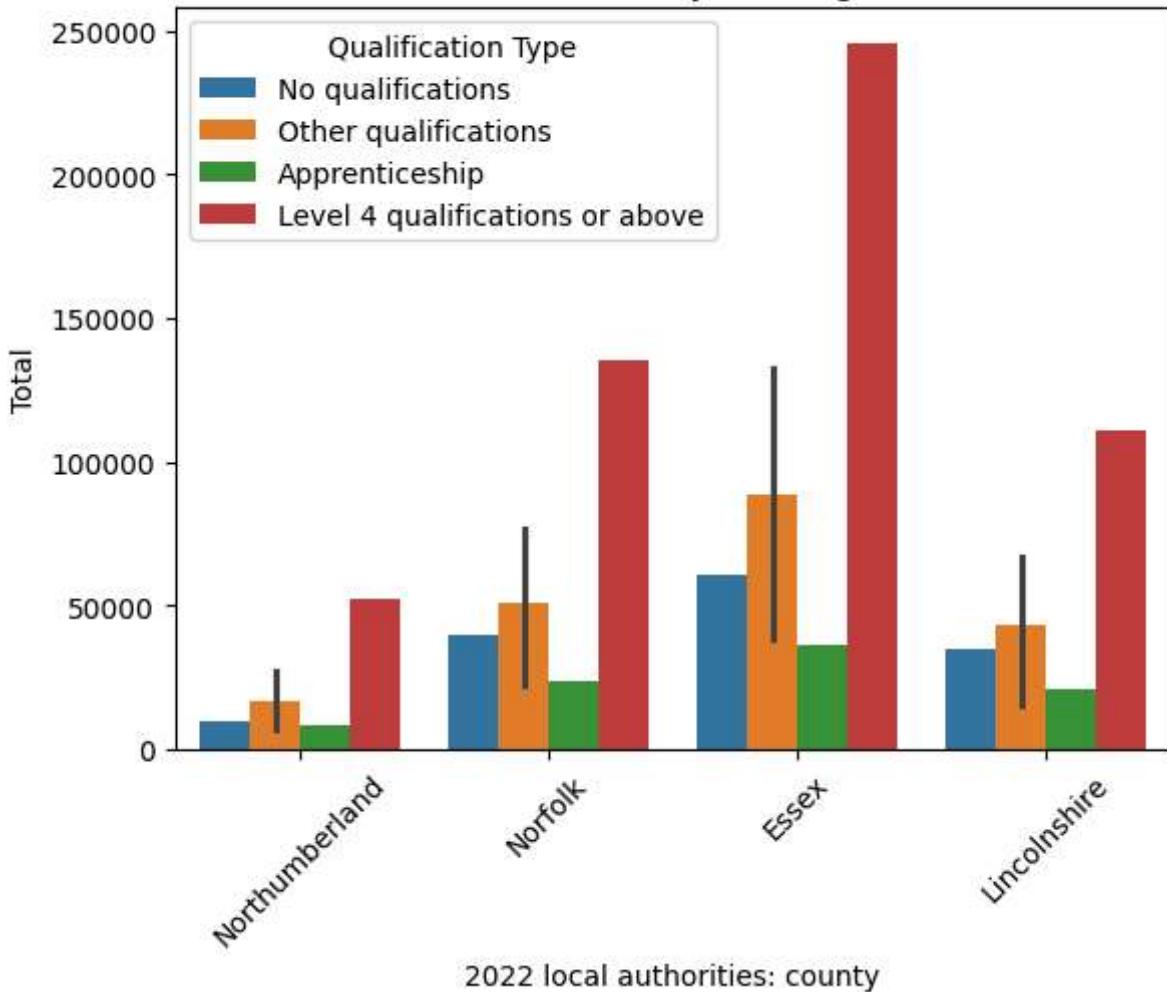
```
Out[37]: ([0, 1, 2],
[Text(0, 0, 'Merton'),
 Text(1, 0, 'Richmond upon Thames'),
 Text(2, 0, 'Wandsworth')])
```



```
In [38]: sns.barplot(data=Bottom5rt, x='2022 local authorities: county', y='Total', hue='Qua  
plt.xticks(rotation=45)
```

```
Out[38]: ([0, 1, 2, 3],  
 [Text(0, 0, 'Northumberland'),  
  Text(1, 0, 'Norfolk'),  
  Text(2, 0, 'Essex'),  
  Text(3, 0, 'Lincolnshire')])
```

Bottom 5 Areas by Ranking Total



```
In [39]: Activity_mapping = {"Employed": 4, "Further Study": 3, "Mixture": 2, "No employment": 1}
GradData['Activity'] = GradData['Activity'].map(Activity_mapping)
```

```
MoFS_mapping = {"Full-time": 3, "Part-time": 2, "All": 1}
GradData['Mode of former study'] = GradData['Mode of former study'].map(MoFS_mapping)
```

```
LoQO_mapping = {"Postgraduate": 3, "Undergraduate": 2, "All": 1}
GradData['Level of qualification obtained'] = GradData['Level of qualification obtained'].map(LoQO_mapping)
```

```
In [40]: GradData = GradData.drop(['Provider name'], axis = 1)
```

```
In [41]: from sklearn.model_selection import train_test_split
```

```
predictors = GradData.drop(['Level of qualification obtained', 'Activity'], axis = 1)
target = GradData["Total"]
x_train, x_val, y_train, y_val = train_test_split(predictors, target, test_size = 0.2, random_state = 42)
```

```
In [42]: from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

```
logreg = LogisticRegression()
logreg.fit(x_train, y_train)
y_pred = logreg.predict(x_val)
acc_logreg = round(accuracy_score(y_pred, y_val) * 100, 2)
print(acc_logreg)
```

70.48

```
In [43]: from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

dt = DecisionTreeClassifier()
dt.fit(x_train, y_train)
y_pred = dt.predict(x_val)
acc_dt = round(accuracy_score(y_pred, y_val) * 100, 2) #Compare The Predicted Value
print(acc_dt)
```

99.98

```
In [44]: from sklearn.ensemble import RandomForestClassifier

randf = RandomForestClassifier()
randf.fit(x_train, y_train)
y_pred = randf.predict(x_val)
acc_randf = round(accuracy_score(y_pred, y_val) * 100, 2) #Compare The Predicted Value
print(acc_randf)
```

99.95

```
In [45]: from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier()
knn.fit(x_train, y_train)
y_pred = knn.predict(x_val)
acc_knn = round(accuracy_score(y_pred, y_val) * 100, 2) #Compare The Predicted Value
print(acc_knn)
```

99.94

```
In [46]: models = pd.DataFrame({
    'Model': ['Logistical Regression', 'Decission Tree', 'Randon Forest', 'KNN'],
    'Score': [acc_logreg, acc_dt, acc_randf, acc_knn]})
```

models.sort_values(by = 'Score', ascending = False)

	Model	Score
1	Decission Tree	99.98
2	Randon Forest	99.95
3	KNN	99.94
0	Logistical Regression	70.48