```
from google.colab import drive
drive.mount('/content/gdrive/', force_remount=True)
```

```
Mounted at /content/gdrive/
```

```
#pip install pyreadstat
```

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statistics
import pandas as pd
import pyreadstat as pr
import math
from math import exp
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.model_selection import LeaveOneOut
from sklearn.metrics import accuracy_score
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_classif
from sklearn import linear_model
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, roc_curve
```

```
#Load London District codes.csv file
df_district = pd.read_csv("/content/gdrive/MyDrive/datasets_udate/raw/dist_code.csv
#print all the data
df_district
```

	District	Districtcode
0	Barking and Dagenham	00AB
1	Barnet	00AC
2	Bexley	00AD
3	Brent	00AE
4	Bromley	00AF
5	Camden	00AG
6	Croydon	00AH
7	Ealing	00AJ
8	Enfield	00AK
9	Greenwich	00AL
10	Hackney	00AM
11	Hammersmith and Fulham	00AN
12	Haringey	00AP
13	Harrow	00AQ
14	Havering	00AR
15	Hillingdon	00AS
16	Hounslow	00AT
17	Islington	00AU
18	Kensington and Chelsea	00AW
19	Kingston upon Thames	00AX
20	Lambeth	00AY

```
#Check duplication
duplicate_district = df_district[df_district.duplicated()]
duplicate_district
```

	District	Districtcode
25	Richmond upon Thames	00BD

Since there is no duplicate row in District data, we can proceed to load the other data.

27	Sutton	00BF

```
#Load London ward data environment.csv
df_ward_environment = pd.read_csv("/content/gdrive/MyDrive/datasets_udate/raw/env.c
df_ward_environment.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 624 entries, 0 to 623
Data columns (total 4 columns):
```

```
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Wardcode              624 non-null    object 
 1   Population2011Census  624 non-null    int64  
 2   Crimerate             624 non-null    float64
 3   Openspace             624 non-null    float64
dtypes: float64(2), int64(1), object(1)
memory usage: 19.6+ KB
```

```
#Check duplication
duplicate_environment = df_ward_environment[df_ward_environment.duplicated()]
duplicate_environment
```

	Wardcode	Population2011Census	Crimerate	Openspace

```
#Check column completeness
print((df_ward_environment['Wardcode'].values == '').sum())
print((df_ward_environment['Population2011Census'].values == 0).sum())
print((df_ward_environment['Crimerate'].values == 0).sum())
print((df_ward_environment['Openspace'].values == 0).sum())
```

```
#Check the Openspace column with 0 value
open_null = df_ward_environment.query("Openspace == 0", engine="python")
open_null
```

	Wardcode	Population2011Census	Crimerate	Openspace
0	00ANGQ	11201	117.7	0.0

As it is possible for a ward not to have an open space, so we are not going to impute the value.

```
#Print the first five data
df_ward_environment.head()
```

	Wardcode	Population2011Census	Crimerate	Openspace
0	00ANGQ	11201	117.7	0.0
1	00ANGA	11518	114.0	0.3
2	00ADGN	10800	44.2	0.7
3	00BEGH	12321	65.3	0.7
4	00BCFZ	12609	64.3	1.3

```
#Since we are going to proceed further with the Wardcode data, then check if there
df_ward_environment.Wardcode.astype(str).str.len().unique()
```

```
array([6, 8])
```

```
#As it can be seen from df_ward_environment.head(), the sampel shows that each Ward
#So we have to find out the one with 8 length
anomaly = df_ward_environment.query("Wardcode.str.len() == 8", engine="python")
anomaly
```

	Wardcode	Population2011Census	Crimerate	Openspace
126	00BAGDag	9816	61.0	12.3

```
#Remove the ag chars from the 00BAGDag so it became 00BAGD
df_ward_environment['Wardcode'] = df_ward_environment['Wardcode'].str.replace('00BA
```

```
#Load London ward data socioeconomic.sav file
df_ward_socioeconomic=pd.read_spss('/content/gdrive/MyDrive/datasets_udate/raw/soci
df_ward_socioeconomic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 657 entries, 0 to 656
Data columns (total 5 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Wardcode          657 non-null    object 
 1   hhSocialRented    621 non-null    float64
 2   JobSeekers        621 non-null    float64
 3   Noqual            621 non-null    float64
 4   Carsperhousehold  621 non-null    float64
dtypes: float64(4), object(1)
memory usage: 25.8+ KB
```

```
#Check duplication
duplicate_socioeconomic = df_ward_socioeconomic[df_ward_socioeconomic.duplicated()]
duplicate_socioeconomic
```

	Wardcode	hhSocialRented	JobSeekers	Noqual	Carsperhousehold
622		NaN	NaN	NaN	NaN
623		NaN	NaN	NaN	NaN
624		NaN	NaN	NaN	NaN
625		NaN	NaN	NaN	NaN
626		NaN	NaN	NaN	NaN
627		NaN	NaN	NaN	NaN
628		NaN	NaN	NaN	NaN
629		NaN	NaN	NaN	NaN
630		NaN	NaN	NaN	NaN
631		NaN	NaN	NaN	NaN
632		NaN	NaN	NaN	NaN
633		NaN	NaN	NaN	NaN
634		NaN	NaN	NaN	NaN
635		NaN	NaN	NaN	NaN
636		NaN	NaN	NaN	NaN
637		NaN	NaN	NaN	NaN
638		NaN	NaN	NaN	NaN
639		NaN	NaN	NaN	NaN
640		NaN	NaN	NaN	NaN
641		NaN	NaN	NaN	NaN
642		NaN	NaN	NaN	NaN
643		NaN	NaN	NaN	NaN
644		NaN	NaN	NaN	NaN

```
#Check column completeness
print((df_ward_socioeconomic['Wardcode'].values == '').sum())
print((df_ward_socioeconomic['hhSocialRented'].values == 0).sum())
print((df_ward_socioeconomic['JobSeekers'].values == 0).sum())
print((df_ward_socioeconomic['Noqual'].values == 0).sum())
print((df_ward_socioeconomic['Carsperhousehold'].values == 0).sum())
```

```
36
0
0
0
0
```

652		NaN	NaN	NaN	NaN

```
#Remove row which does not have wardcode
df_ward_socioeconomic = df_ward_socioeconomic[df_ward_socioeconomic.Wardcode != '']
```

654		NaN	NaN	NaN	NaN

```
#Since we are going to proceed further with the Wardcode data, then check if there
df_ward_socioeconomic.Wardcode.astype(str).str.len().unique()
```

```
array([6, 7])
```

```
#As it can be seen from df_ward_environment.head(), the sampel shows that each Ward
#So we have to find out the one with 8 length
anomaly = df_ward_socioeconomic.query("Wardcode.str.len() == 7", engine="python")
anomaly
```

	Wardcode	hhSocialRented	JobSeekers	Noqual	Carsperhousehold
107	00AGGK#	45.9	6.3	17.8	0.5

```
#Remove the # char from the 00AGGK# so it became 00AGGK
df_ward_socioeconomic['Wardcode'] = df_ward_socioeconomic['Wardcode'].str.replace(
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: SettingWithCo
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/s
```

```
#Check how many Wardcode are not the same from Socioeconomic and Environment
check_wardcode = df_ward_socioeconomic[~df_ward_socioeconomic['Wardcode'].isin(df_w
check_wardcode
```

	Wardcode	hhSocialRented	JobSeekers	Noqual	Carsperhousehold

As it can be seen, that the Wardcode from Socioeconomic data are precisely similar with those from Environment data.

```
#Combine Socioeconomic and Environment by Wardcode
df_merge_ward_environment_socioeconomic = pd.merge(df_ward_environment, df_ward_soc
df_merge_ward_environment_socioeconomic
```

	Wardcode	Population2011Census	Crimerate	Openspace	hhSocialRented	Job
0	00ANGQ	11201	117.7	0.0	23.7	
1	00ANGA	11518	114.0	0.3	24.9	
2	00ADGN	10800	44.2	0.7	1.1	
3	00BEGH	12321	65.3	0.7	20.3	
4	00BCFZ	12609	64.3	1.3	4.7	
...	...	...	...	...	...	...

```
#Load london ward data health.sas7bdat file
df_ward_data_health, meta = pr.read_sas7bdat('/content/gdrive/MyDrive/datasets_udat
df_ward_data_health.head()
```

	Wardname	Population2011Census	GeneralFertilityR
0	Bromley - Darwin	5110.0	
1	Kensington and Chelsea - Royal Hospital	7252.0	
2	Hillingdon - Harefield	7399.0	
3	Hammersmith and Fulham - Palace Riverside	7483.0	
4	Kensington and Chelsea - Pembridge	7659.0	

```
#Check column completeness
print((df_ward_data_health['Wardname'].values == '').sum())
print((df_ward_data_health['Population2011Census'].values == 0).sum())
print((df_ward_data_health['GeneralFertilityRate'].values == 0).sum())
print((df_ward_data_health['Malelifeexpectancy'].values == 0).sum())
print((df_ward_data_health['Femalelifeexpectancy'].values == 0).sum())
```

```
0
0
0
0
0
```

```
#Check duplicate data
duplicate_health = df_ward_data_health[df_ward_data_health.duplicated()]
duplicate_health
```

	Wardname	Population2011Census	GeneralFertilityRate	Malelifeexpectancy	Fe

```
#Check the Wardname column unique value
#df_ward_data_health['Wardname'] = df_ward_data_health['Wardname'].astype('str').st
df_ward_data_health['Wardname'].unique()
```

```
array(['Bromley - Darwin', 'Kensington and Chelsea - Royal Hospital',
       'Hillingdon - Harefield',
       'Hammersmith and Fulham - Palace Riverside',
       'Kensington and Chelsea - Pembridge',
       'Kensington and Chelsea - Cremorne', 'Westminster - Tachbrook',
       'Kensington and Chelsea - Campden',
       'Kensington and Chelsea - Stanley',
       'Kensington and Chelsea - Colville', 'Merton - Village',
       'Kensington and Chelsea - Norland',
       'Kingston upon Thames - Chessington North and Hook',
       'Kensington and Chelsea - Hans Town',
       'Kensington and Chelsea - Brompton',
       'Kensington and Chelsea - Courtfield',
       'Kensington and Chelsea - Golborne',
       'Kingston upon Thames - St. James', 'Merton - Hillside',
       'Merton - Lower Morden', "Kensington and Chelsea - Earl's Court",
       'Hammersmith and Fulham - College Park and Old Oak',
       'Kensington and Chelsea - Redcliffe',
       'Kingston upon Thames - Alexandra', 'Merton - Cannon Hill',
       'Westminster - Knightsbridge and Belgravia', 'Merton - Dundonald',
       'Kensington and Chelsea - St. Charles', 'Westminster - Warwick',
       'Richmond upon Thames - Hampton North',
       'Kingston upon Thames - Old Malden',
       'Kingston upon Thames - Berrylands', 'Merton - Merton Park',
       'Kensington and Chelsea - Notting Barns',
       'Kingston upon Thames - Tudor', 'Sutton - Sutton South',
       'Westminster - Churchill', "Lambeth - Bishop's",
       'Sutton - Carshalton South and Clockhouse', 'Merton - Raynes
Park',
       'Kingston upon Thames - Coombe Vale',
       'Richmond upon Thames - Whitton',
       'Kensington and Chelsea - Holland', 'Merton - Trinity',
       'Merton - Graveney', 'Bromley - Shortlands',
       'Kingston upon Thames - Tolworth and Hook Rise',
       'Barking and Dagenham - Parsloes',
       "Kensington and Chelsea - Queen's Gate", 'Merton - West Barnes',
       'Barking and Dagenham - Valence', 'Bromley - Biggin Hill',
       'Merton - Ravensbury', 'Richmond upon Thames - Hampton',
       'Richmond upon Thames - South Twickenham',
       'Bromley - Mottingham and Chislehurst North',
       'Westminster - Vincent Square',
       'Kensington and Chelsea - Abingdon',
       'Barking and Dagenham - Chadwell Heath', 'Merton - Longthornton',
       'Harrow - Pinner', 'Sutton - Carshalton Central',
       'Sutton - Belmont', 'Harrow - Headstone North',
       'Kingston upon Thames - Norbiton',
       'Kingston upon Thames - Beverley',
       'Richmond upon Thames - Fulwell and Hampton Hill',
       'Sutton - The Wrythe', 'Sutton - Wallington South',
       'Westminster - Maida Vale', 'Richmond upon Thames - Hampton Wick',
       'Kingston upon Thames - Chessington South',
```

```
       'Merton - Lavender Fields', 'Sutton - Cheam',
       'Tower Hamlets - Spitalfields and Banglatown',
       'Merton - Pollards Hill', 'Waltham Forest - Chingford Green',
       'Richmond upon Thames - Barnes', 'Westminster - Bayswater',
       'Sutton - Beddington North', 'Richmond upon Thames - Heathfield',
       'Richmond upon Thames - Ham, Petersham & Richmond Riverside',
```

```
#Load London ward data demographics.dat file
df_demographics = pd.read_csv('/content/gdrive/MyDrive/datasets_udate/raw/demo.dat
df_demographics.head()
```

	Wardname	Children	Greaterthan65	nonwhite	NotBornir
0	\nHackney - Queensbridge	17.542063	8.2	44.7	3
1	\nHammersmith & Fulham - S&s End	17.915361	8.4	30.6	3
2	\nBarking & Dagenham - River	26.851598	10.1	38.4	3
3	\nTower Hamlets - Bethnal Green North	18.555872	7.0	49.4	3
4	\nMerton - Abbey	15.731861	8.5	26.6	3

```
#Check column completeness
print((df_demographics['Wardname'].values == '').sum())
print((df_demographics['Greaterthan65'].values == 0).sum())
print((df_demographics['nonwhite'].values == 0).sum())
print((df_demographics['NotBorninUK'].values == 0).sum())
print((df_demographics['NotEnglishspeaking'].values == 0).sum())
```

```
0
0
0
0
0
```

```
#It is obvious that there is \n char in front of every Wardname, so we must remove
df_demographics['Wardname'] = df_demographics['Wardname'].str.replace('\n','')
```

```
df_demographics['Wardname'].unique()
```

```
array(['Hackney - Queensbridge', 'Hammersmith & Fulham - S&s End',
       'Barking & Dagenham - River',
       'Tower Hamlets - Bethnal Green North', 'Merton - Abbey',
       'Richmond upon Thames - South Richmond', 'Lambeth - Larkhall',
       'Greenwich - Plumstead', 'Hillingdon - Heathrow Villages',
       'Lambeth - Streatham Hill', 'Havering - Gooshays',
       'Croydon - Broad Green', 'Hillingdon - West Ruislip',
       'Ealing - North Greenford', 'Lewisham - Lewisham Central',
       'Hillingdon - Hillingdon East', 'Wandsworth - Southfields',
       'Hounslow - Heston West', 'Sutton - Sutton West',
       'Merton - Dundonald', 'Haringey - West Green',
       'Hounslow - Heston Central', 'Barnet - Underhill',
       'Sutton - Sutton Central', 'Bexley - Lesnes Abbey',
```

```
       'Hammersmith & Fulham - Hammersmith Broadway',
       'Southwark - Brunswick Park', 'Redbridge - Clementswood',
       'Islington - Barnsbury', 'Croydon - Shirley',
       'Wandsworth - Northcote', 'Greenwich - Woolwich Common',
       'Croydon - Sanderstead', 'Croydon - West Thornton',
       'Barking & Dagenham - Valence', 'Harrow - Harrow Weald',
       'Lewisham - Lee Green', 'Kensington & Chelsea - Royal Hospital',
       'Lambeth - Thornton', 'Harrow - Greenhill', 'Bexley - Sidcup',
       'Lambeth - Herne Hill', 'Kensington & Chelsea - Brompton',
       "Merton - Figge's Marsh", 'Redbridge - Fairlop',
       'Greenwich - Middle Park & Sutcliffe',
       'Richmond upon Thames - East Sheen', 'Merton - Wimbledon Park',
       'Barking & Dagenham - Parsloes', 'Kensington & Chelsea - Campden',
       'Islington - Tollington', 'Richmond upon Thames - Hampton',
       'Greenwich - Thamesmead Moorings', 'Enfield - Lower Edmonton',
       'Brent - Preston', 'Wandsworth - Roehampton', 'Harrow - Edgware',
       'Havering - Harold Wood', 'Hillingdon - Townfield',
       'Hillingdon - Northwood', 'Kingston upon Thames - Alexandra',
       'Bexley - Crayford', 'Sutton - Wandle Valley',
       'Ealing - Dormers Wells', 'Newham - Canning Town South',
       'Hackney - Dalston', 'Haringey - Seven Sisters',
       'Newham - Green Street West', 'Bromley - Cray Valley West',
       'Islington - Bunhill', "Lambeth - Bishop's", 'Hillingdon -
Brunel',
       'Bexley - Longlands', 'Croydon - Upper Norwood',
       "Camden - Regent's Park", 'Hillingdon - Uxbridge North',
       'Barnet - Finchley Church End', "Lambeth - Knight's Hill",
       'Lambeth - Gipsy Hill', 'Harrow - Canons', 'Redbridge - Roding',
       "Islington - Saint George's",
       'Barking & Dagenham - Chadwell Heath', 'Redbridge - Barkingside',
       'Hackney - Stoke Newington Central',
       'Kingston upon Thames - Coombe Hill', 'Sutton - Beddington South',
       'Harrow - Harrow on the Hill', 'Ealing - Norwood Green',
       'Richmond upon Thames - North Richmond',
       'Enfield - Upper Edmonton', 'Lewisham - Rushey Green',
       'Kingston upon Thames - Chessington North & Hook',
       'Havering - Hylands', "Westminster - Queen's Park",
       'Kingston upon Thames - Chessington South',
       'Wandsworth - Shaftesbury', 'Hillingdon - Eastcote & East
Ruislip',
       'Hammersmith & Fulham - Palace Riverside', 'Redbridge - Fullwell',
       'Westminster - Churchill', 'Westminster - Warwick',
       'Merton - Raynes Park', 'Kensington & Chelsea - Redcliffe',
       'Lambeth - Ferndale', 'Croydon - Ashburton',
       'Tower Hamlets - Limehouse', 'Camden - Cantelowes',
```

```
#It is obvious that there are two different chars or symbols between Wardname in He
#First, in Demograhic, it uses "&" instead of "and", as in District data it uses "a
df_demographics['Wardname'] = df_demographics['Wardname'].astype('str').str.replace
#Second, in Demographic, it uses "Saint", and in Health it uses St., so we replace
df_demographics['Wardname'] = df_demographics['Wardname'].str.replace('Saint','St.
#Check if it has been replaced
df_demographics['Wardname'].unique()
```

```
array(['Hackney - Queensbridge', 'Hammersmith and Fulham - Sands End',
       'Barking and Dagenham - River',
       'Tower Hamlets - Bethnal Green North', 'Merton - Abbey',
       'Richmond upon Thames - South Richmond', 'Lambeth - Larkhall',
       'Greenwich - Plumstead', 'Hillingdon - Heathrow Villages',
```

```
            'Lambeth - Streatham Hill', 'Havering - Gooshays',
            'Croydon - Broad Green', 'Hillingdon - West Ruislip',
            'Ealing - North Greenford', 'Lewisham - Lewisham Central',
            'Hillingdon - Hillingdon East', 'Wandsworth - Southfields',
            'Hounslow - Heston West', 'Sutton - Sutton West',
            'Merton - Dundonald', 'Haringey - West Green',
            'Hounslow - Heston Central', 'Barnet - Underhill',
            'Sutton - Sutton Central', 'Bexley - Lesnes Abbey',
            'Hammersmith and Fulham - Hammersmith Broadway',
            'Southwark - Brunswick Park', 'Redbridge - Clementswood',
            'Islington - Barnsbury', 'Croydon - Shirley',
            'Wandsworth - Northcote', 'Greenwich - Woolwich Common',
            'Croydon - Sanderstead', 'Croydon - West Thornton',
            'Barking and Dagenham - Valence', 'Harrow - Harrow Weald',
            'Lewisham - Lee Green', 'Kensington and Chelsea - Royal Hospital',
            'Lambeth - Thornton', 'Harrow - Greenhill', 'Bexley - Sidcup',
            'Lambeth - Herne Hill', 'Kensington and Chelsea - Brompton',
            "Merton - Figge's Marsh", 'Redbridge - Fairlop',
            'Greenwich - Middle Park and Sutcliffe',
            'Richmond upon Thames - East Sheen', 'Merton - Wimbledon Park',
            'Barking and Dagenham - Parsloes',
            'Kensington and Chelsea - Campden', 'Islington - Tollington',
            'Richmond upon Thames - Hampton',
            'Greenwich - Thamesmead Moorings', 'Enfield - Lower Edmonton',
            'Brent - Preston', 'Wandsworth - Roehampton', 'Harrow - Edgware',
            'Havering - Harold Wood', 'Hillingdon - Townfield',
            'Hillingdon - Northwood', 'Kingston upon Thames - Alexandra',
            'Bexley - Crayford', 'Sutton - Wandle Valley',
            'Ealing - Dormers Wells', 'Newham - Canning Town South',
            'Hackney - Dalston', 'Haringey - Seven Sisters',
            'Newham - Green Street West', 'Bromley - Cray Valley West',
            'Islington - Bunhill', "Lambeth - Bishop's", 'Hillingdon -
        Brunel',
            'Bexley - Longlands', 'Croydon - Upper Norwood',
            "Camden - Regent's Park", 'Hillingdon - Uxbridge North',
            'Barnet - Finchley Church End', "Lambeth - Knight's Hill",
            'Lambeth - Gipsy Hill', 'Harrow - Canons', 'Redbridge - Roding',
            "Islington - St. George's",
            'Barking and Dagenham - Chadwell Heath', 'Redbridge -
        Barkingside',
            'Hackney - Stoke Newington Central',
            'Kingston upon Thames - Coombe Hill', 'Sutton - Beddington South',
            'Harrow - Harrow on the Hill', 'Ealing - Norwood Green',
            'Richmond upon Thames - North Richmond',
            'Enfield - Upper Edmonton', 'Lewisham - Rushey Green',
            'Kingston upon Thames - Chessington North and Hook',
            'Havering - Hylands', "Westminster - Queen's Park",
            'Kingston upon Thames - Chessington South',
            'Wandsworth - Shaftesbury',
            'Hillingdon - Eastcote and East Ruislip',
            'Hammersmith and Fulham - Palace Riverside',
            'Redbridge - Fullwell', 'Westminster - Churchill',
            'Westminster - Warwick', 'Merton - Raynes Park',
```

```
#Check how many Wardcode are not the same from Health and Demographic
check_wardname = df_ward_data_health[~df_ward_data_health['Wardname'].isin(df_demog
check_wardname
```

	Wardname	Population2011Census	General
67	Sutton - The Wrythe	10163.0	
81	Richmond upon Thames - Ham, Petersham & Richmo...	10317.0	
153	Richmond upon Thames - St. Margarets & North T...	11172.0	

```
#Combine Socioeconomic and Environment by Wardname
df_merge_ward_health_demographics = pd.merge(df_ward_data_health, df_demographics,
df_merge_ward_health_demographics
```

	Wardname	Population2011Census	GeneralFertilityRate	Malelifeexpectan
0	Bromley - Darwin	5110.0	63.8	8
1	Kensington and Chelsea - Royal Hospital	7252.0	52.3	8(
2	Hillingdon - Harefield	7399.0	55.8	7

```
#As we will use the Population Cencus later to combine values, we have to drop thos
df_merge_ward_health_demographics = df_merge_ward_health_demographics.dropna()
df_merge_ward_health_demographics
```

	Wardname	Population2011Census	GeneralFertilityRate	Malelifeexpectan

```
#Since we have the district and district code only as reference, we try to get dist
df_merge_ward_health_demographics[['District','Ward']] = df_merge_ward_health_demog
#As we only need the district, we can drop the Ward
df_merge_ward_health_demographics = df_merge_ward_health_demographics.drop('Ward',
df_merge_ward_health_demographics['District'] = df_merge_ward_health_demographics[
#Print the district
df_merge_ward_health_demographics
```

```
/usr/local/lib/python3.7/dist-packages/pandas/core/frame.py:3641: SettingWith
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
```

```
df_district['District'] = df_district['District'].astype('str').str.replace(' ','')
```

```
#Check how many District are not the same from Health and Demographic data compare
check_district = df_merge_ward_health_demographics[~df_merge_ward_health_demographi
check_district
```

	Wardname	Population2011Census	GeneralFertilityRate	Malelifeexpectancy	Fe

```
df_merge_ward_health_demographics_district = pd.merge(df_merge_ward_health_demograp
#df_merge_ward_health_demographics_district = df_merge_ward_health_demographics_dis
df_merge_ward_health_demographics_district
```

Now we have two data that have not been linked. We will link them using a new ID called MyID that combine District Code and its population.

```
#First, create MyID from Environment and Socioeconomic data
#Create the District code column, take the first four digit from Wardcode column
df_merge_ward_environment_socioeconomic['Districtcode'] = df_merge_ward_environment
#Create MyID Column by combining District code and Population cencus
df_merge_ward_environment_socioeconomic['MyID'] = df_merge_ward_environment_socioec
df_merge_ward_environment_socioeconomic['MyID'] = df_merge_ward_environment_socioec
df_merge_ward_environment_socioeconomic['MyID'] = df_merge_ward_environment_socioec
df_merge_ward_environment_socioeconomic
```

	Wardcode	Population2011Census	Crimerate	Openspace	hhSocialRented	Job
0	00ANGQ	11201	117.7	0.0	23.7	
1	00ANGA	11518	114.0	0.3	24.9	
2	00ADGN	10800	44.2	0.7	1.1	
3	00BEGH	12321	65.3	0.7	20.3	
4	00BCFZ	12609	64.3	1.3	4.7	
...	...	...	...	...	...	
619	00BDFZ	10317	65.8	81.9	18.5	
620	00ARGW	12833	49.5	82.0	1.9	
621	00ASGN	7399	67.4	85.5	24.0	
622	00AFGQ	5110	58.6	88.8	5.5	
623	00AZGW	16414	71.7	99.9	36.6	

624 rows × 10 columns

```
#And then, create MyID from Health and Demographic data
#Create MyID Column by combining District code and Population cencus
df_merge_ward_health_demographics_district['MyID'] = df_merge_ward_health_demograph
df_merge_ward_health_demographics_district['MyID'] = df_merge_ward_health_demograph
df_merge_ward_health_demographics_district
```

	Wardname	Population2011Census	GeneralFertilityRate	Malelifeexpectancy
0	Bromley - Darwin	5110.0	63.8	81.2
1	Bromley - Shortlands	9824.0	48.2	83.0
2	Bromley - Biggin Hill	9951.0	60.0	82.2
3	Bromley - Mottingham and Chislehurst North	9987.0	70.0	76.6
4	Bromley - Crystal Palace	12255.0	70.6	75.5
...	...	...	...	..
614	Barnet - Burnt Oak	18217.0	94.3	74.7
615	Barnet - Mill Hill	18451.0	63.8	80.7
616	Barnet - Hendon	18472.0	88.4	80.1

```
#Check how many Wardcode are the same from Socioeconomic and Environment compare to
test = df_merge_ward_environment_socioeconomic[~df_merge_ward_environment_socioecor
test
```

	Wardcode	Population2011Census	Crimerate	Openspace	hhSocialRented	Job
138	00AYGN	14777	94.4	13.2	45.0	
229	00BFGR	10163	56.2	18.0	14.7	
235	00BCGL	16544	108.0	18.3	22.8	
309	00BDGH	11172	55.1	23.5	4.0	
619	00BDFZ	10317	65.8	81.9	18.5	

```
#Combine the data so it is integrated
df_merge_all = pd.merge(df_merge_ward_environment_socioeconomic, df_merge_ward_heal
df_merge_all.head()
```

	Wardcode	Population2011Census_x	Crimerate	Openspace	hhSocialRented	Job
0	00ANGQ	11201	117.7	0.0	23.7	
1	00ANGA	11518	114.0	0.3	24.9	
2	00ADGN	10800	44.2	0.7	1.1	
3	00BEGH	12321	65.3	0.7	20.3	

```
df_merge_all = df_merge_all.dropna()
df_merge_all
```

	Wardcode	Population2011Census_x	Crimerate	Openspace	hhSocialRented	J
0	00ANGQ	11201	117.7	0.0	23.7	
1	00ANGA	11518	114.0	0.3	24.9	
2	00ADGN	10800	44.2	0.7	1.1	
3	00BEGH	12321	65.3	0.7	20.3	
4	00BCFZ	12609	64.3	1.3	4.7	
...	...	...	...	...	...	
618	00AKGN	13762	52.5	81.5	21.3	
620	00ARGW	12833	49.5	82.0	1.9	
621	00ASGN	7399	67.4	85.5	24.0	
622	00AFGQ	5110	58.6	88.8	5.5	
623	00AZGW	16414	71.7	99.9	36.6	

616 rows × 22 columns

```
df_merge_all = df_merge_all.drop(['Population2011Census_x','Districtcode_x','MyID',
df_merge_all
```

	Wardcode	Crimerate	Openspace	hhSocialRented	JobSeekers	Noqual	Carspe
0	00ANGQ	117.7	0.0	23.7	3.6	9.4	
1	00ANGA	114.0	0.3	24.9	4.5	9.3	
2	00ADGN	44.2	0.7	1.1	2.9	22.1	
3	00BEGH	65.3	0.7	20.3	4.6	12.3	
4	00BCFZ	64.3	1.3	4.7	4.2	17.2	
...	...	...	...	...	...	...	

```
df_merge_all = df_merge_all.rename(columns={"Population2011Census_y": "Population2(
df_merge_all
```

	Wardcode	Crimerate	Openspace	hhSocialRented	JobSeekers	Noqual	Carspe

```
df_merge_all = df_merge_all[['Wardcode', 'Wardname', 'Malelifeexpectancy', 'Femalel
df_merge_all
```

	Wardcode	Wardname	Malelifeexpectancy	Femalelifeexpectancy	Openspace
0	00ANGQ	Hammersmith and Fulham - Town	78.8	81.1	0.(
1	00ANGA	Hammersmith and Fulham - Addison	78.8	88.6	0.3
2	00ADGN	Bexley - Falconwood and Welling	78.2	82.8	0.1
3	00BEGH	Southwark - East Dulwich	81.2	82.9	0.1
4	00BCFZ	Redbridge - Barkingside	81.0	85.7	1.3
...	...	...	...	...	..
618	00AKGN	Enfield - Chase	76.8	80.8	81.5
620	00ARGW	Havering - Upminster	80.1	81.6	82.(
621	00ASGN	Hillingdon - Harefield	78.3	82.3	85.5
622	00AFGQ	Bromley - Darwin	81.2	82.4	88.8
623	00AZGW	Lewisham - Telegraph Hill	76.4	78.8	99.9

616 rows × 16 columns

```
df_male = df_merge_all.drop(['Femalelifeexpectancy','Wardcode','Wardname'],axis=1)
df_male.head()
```

	Malelifeexpectancy	Openspace	hhSocialRented	JobSeekers	Noqual	Carsperh
1	78.8	0.3	24.9	4.5	9.3	

```
#sns.pairplot(df_male)
```

```
df_female = df_merge_all.drop(['Malelifeexpectancy','Wardcode','Wardname'],axis=1)
df_female
```

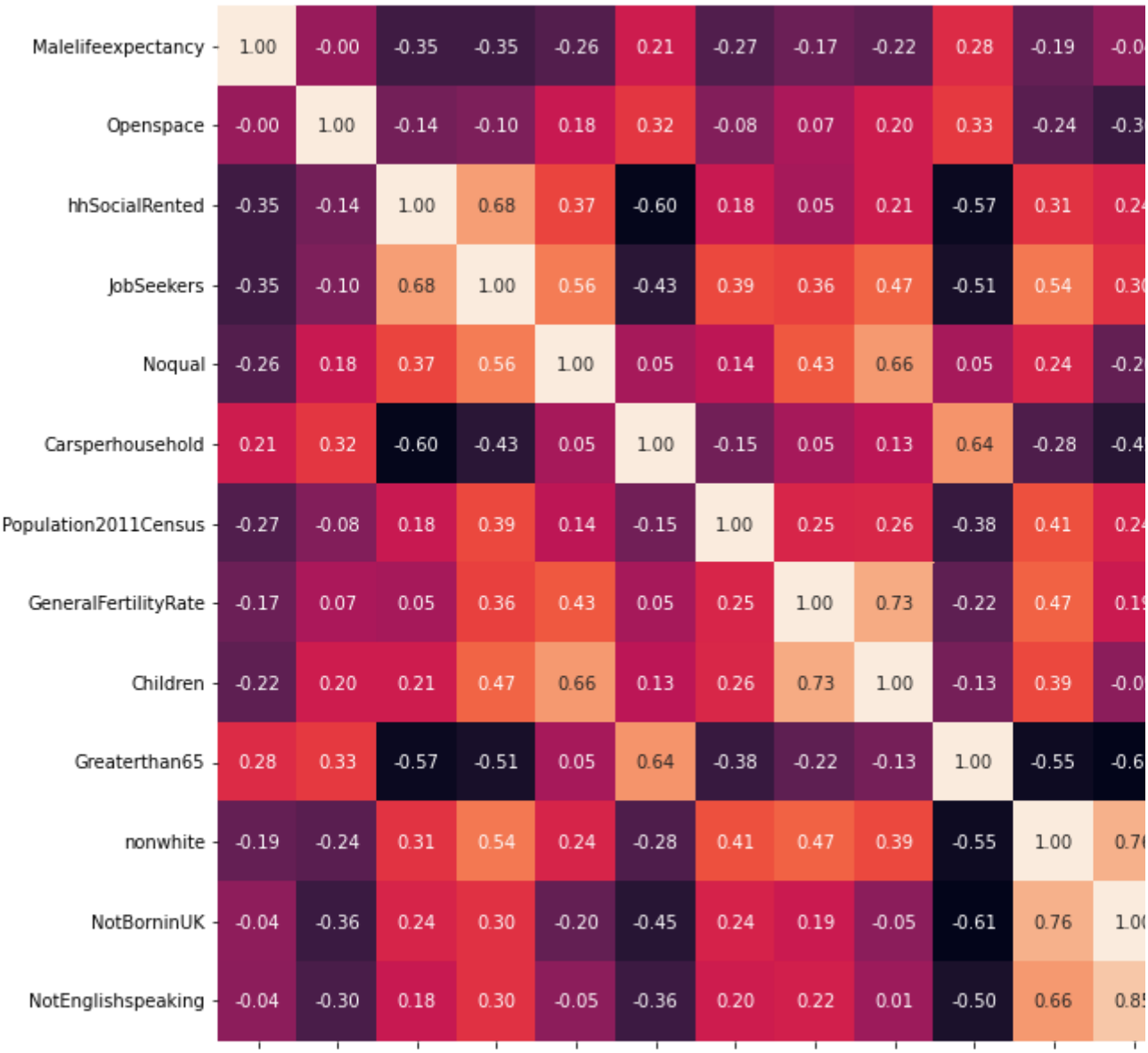
	Femalelifeexpectancy	Openspace	hhSocialRented	JobSeekers	Noqual	Cars
0	81.1	0.0	23.7	3.6	9.4	
1	88.6	0.3	24.9	4.5	9.3	
2	82.8	0.7	1.1	2.9	22.1	
3	82.9	0.7	20.3	4.6	12.3	
4	85.7	1.3	4.7	4.2	17.2	
...	...	...	...	...	...	
618	80.8	81.5	21.3	6.5	22.4	
620	81.6	82.0	1.9	31.6	19.0	
621	82.3	85.5	24.0	3.2	23.8	
622	82.4	88.8	5.5	1.6	21.9	
623	78.8	99.9	36.6	8.2	14.4	

616 rows × 13 columns

```
#sns.pairplot(df_female)
```

```
#Draw Heatmap
corrmat = df_male.corr()

fig, ax = plt.subplots(figsize=(13,13))
hm = sns.heatmap(corrmat,
                 cbar=True,
                 annot=True,
                 square=True,
                 fmt='.2f',
                 yticklabels=df_male.columns,
                 xticklabels=df_male.columns)
plt.show()
```

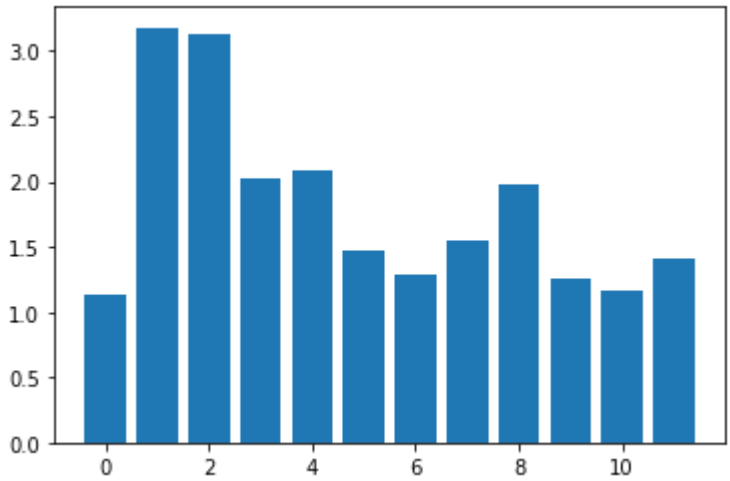



```
#Create function for selecting best features to predict the best model
def select_features(X_train, y_train, X_test):
    # configure to select all features
    fs = SelectKBest(score_func=f_classif, k='all')
    # learn relationship from training data
    fs.fit(X_train, y_train)
    # transform train input data
    X_train_fs = fs.transform(X_train)
    # transform test input data
    X_test_fs = fs.transform(X_test)
    return X_train_fs, X_test_fs, fs


#Select the best features to predict the best model
df_copy = df_male.copy()
df_copy = df_copy.drop(columns=['Malelifeexpectancy'],axis=1)
X = df_copy.values
y = df_male['Malelifeexpectancy'].values
X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=0.1, random_stat
```

```
X_train_fs, X_test_fs, fs = select_features(X_train, y_train, X_test)
# what are scores for the features
for i in range(len(fs.scores_)):
    print('Feature %d: %f' % (i, fs.scores_[i]))
# plot the scores
plt.bar([i for i in range(len(fs.scores_))], fs.scores_)
plt.show()
```

```
Feature 0: 1.127420
Feature 1: 3.180103
Feature 2: 3.131653
Feature 3: 2.031203
Feature 4: 2.083877
Feature 5: 1.475490
Feature 6: 1.292965
Feature 7: 1.548294
Feature 8: 1.978494
Feature 9: 1.250634
Feature 10: 1.159946
Feature 11: 1.409043
```




```
from sklearn.metrics import mean_squared_error, r2_score


# Train-test split
X_train2, X_test2, y_train2, y_test2 = train_test_split(X, y,  test_size=0.1, randc

# Builds the classifier
lin2 = LinearRegression()

# Fitting the data
lin2 = lin2.fit(X_train2, y_train2)

# Predicting the data
y_pred2 = lin2.predict(X_test2)

# Evaluates on the test data
print("Mean squared error: %.2f" % mean_squared_error(y_test2, y_pred2))
# The coefficient of determination: 1 is perfect prediction
print("Coefficient of determination: %.2f" % r2_score(y_test2, y_pred2))
```

```
Mean squared error: 6.65
Coefficient of determination: 0.32
```

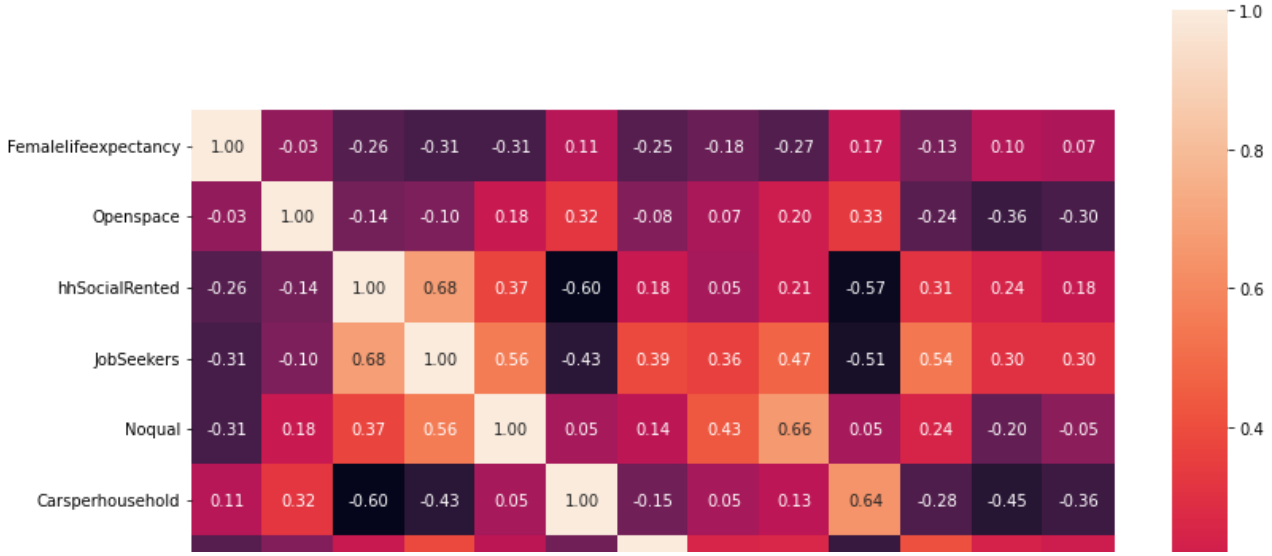
```
from sklearn.ensemble import RandomForestRegressor


Model = RandomForestRegressor(n_estimators=50)
Model = Model.fit(X_train2, y_train2)
y_pred3 = Model.predict(X_test2)
print("Mean squared error: %.2f" % mean_squared_error(y_test2, y_pred3))
# The coefficient of determination: 1 is perfect prediction
print("Coefficient of determination: %.2f" % r2_score(y_test2, y_pred3))
```

```
Mean squared error: 11.91
Coefficient of determination: -0.22
```

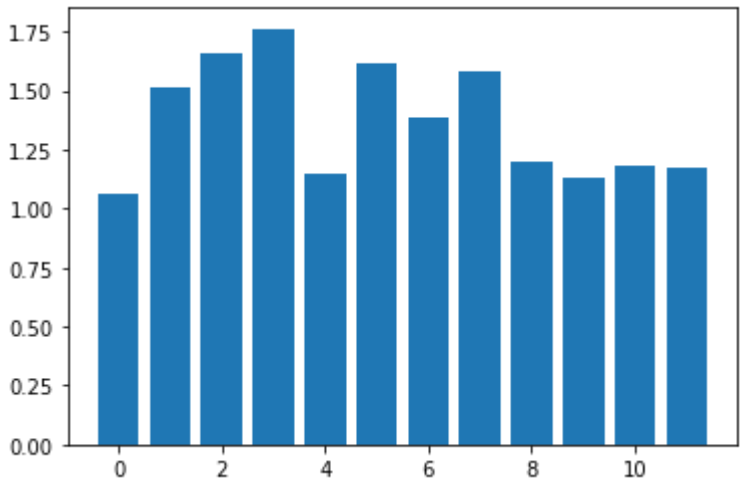
```
#Draw Heatmap
corrmat = df_female.corr()

fig, ax = plt.subplots(figsize=(13,13))
hm = sns.heatmap(corrmat,
                 cbar=True,
                 annot=True,
                 square=True,
                 fmt='.2f',
                 yticklabels=df_female.columns,
                 xticklabels=df_female.columns)
plt.show()
```




```
#Select the best features to predict the best model
df_copy = df_female.copy()
df_copy = df_copy.drop(columns=['Femalelifeexpectancy'],axis=1)
X = df_copy.values
y = df_female['Femalelifeexpectancy'].values
X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=0.1, random_stat
X_train_fs, X_test_fs, fs = select_features(X_train, y_train, X_test)
# what are scores for the features
for i in range(len(fs.scores_)):
    print('Feature %d: %f' % (i, fs.scores_[i]))
# plot the scores
plt.bar([i for i in range(len(fs.scores_))], fs.scores_)
plt.show()
```

```
Feature 0: 1.066263
Feature 1: 1.511521
Feature 2: 1.655550
Feature 3: 1.762516
Feature 4: 1.149859
Feature 5: 1.617367
Feature 6: 1.386646
Feature 7: 1.581244
Feature 8: 1.197759
Feature 9: 1.128701
Feature 10: 1.181468
Feature 11: 1.170013
```



```
# Train-test split
X_train2, X_test2, y_train2, y_test2 = train_test_split(X, y,  test_size=0.1, randc

# Builds the classifier
lin2 = LinearRegression()

# Fitting the data
lin2 = lin2.fit(X_train2, y_train2)

# Predicting the data
y_pred2 = lin2.predict(X_test2)

# Evaluates on the test data
print("Mean squared error: %.2f" % mean_squared_error(y_test2, y_pred2))
# The coefficient of determination: 1 is perfect prediction
print("Coefficient of determination: %.2f" % r2_score(y_test2, y_pred2))
```

```
Mean squared error: 11.62
Coefficient of determination: 0.08
```