logistic-regression-nrcm

August 25, 2023

NAME : A.SUMANTH

ROLL NO :21X05A6704

BRANCH : CSE(DS)

NRCM

PROJRCT TITLE

Prediction of "social_network_ads.csv" data set to estimate the future prediction for "Age" vs "estimated salary"

1 Problem statment

A Insian News chancel "Z24" as predictedsalary estimation for financial year 2018 2019 The organization wants to cut of the "salary" to be safe by impacting huge loss ##Task

As a Data science professional select the particular algorithm and Predict the futurestic Estimated salary.

2 Logistic Regression

2.1 Importing the libraries

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

2.2 Importing the dataset

[ ]:
```
dataset= pd.read_csv("/content/Social_Network_Ads.csv")
dataset
```

[ ]:

	Age	EstimatedSalary	Purchased
0	19	19000	0
1	35	20000	0
2	26	43000	0
3	27	57000	0

```
4      19              76000           0
..    …                …         …
395    46              41000           1
396    51              23000           1
397    50              20000           1
398    36              33000           0
399    49              36000           1

[400 rows x 3 columns]
```

[ ]:

```

```

2.3 Splitting the dataset into the Training set and Test set

[ ]:

```python
x=dataset.iloc[:,:-1].values
y=dataset.iloc[:,-1].values
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split (x,y,test_size=0.
 ↪2,random_state=42)
```

[ ]:

```python
print(x_train)
```

```
[[    27  57000]
 [    46  28000]
 [    39 134000]
 [    44  39000]
 [    57  26000]
 [    32 120000]
 [    41  52000]
 [    48  74000]
 [    26  86000]
 [    22  81000]
 [    49  86000]
 [    36  54000]
 [    40  59000]
 [    41  80000]
 [    26  16000]
 [    39  79000]
 [    59 130000]
 [    42  64000]
 [    53 143000]
 [    34 112000]
 [    57 122000]
 [    39  71000]
 [    47  25000]
 [    24  19000]
 [    36  50000]
```

```
[     32 150000]
[     48  29000]
[     30 107000]
[     60  34000]
[     38  61000]
[     33  31000]
[     39  71000]
[     55  39000]
[     49  39000]
[     43 112000]
[     27  20000]
[     26  17000]
[     37  93000]
[     42  54000]
[     35  61000]
[     29  75000]
[     38  80000]
[     45  26000]
[     54 108000]
[     46  23000]
[     23  28000]
[     37  75000]
[     42  65000]
[     35  71000]
[     51 146000]
[     39  96000]
[     24  89000]
[     58  95000]
[     25  22000]
[     41  59000]
[     28  89000]
[     42  80000]
[     42 108000]
[     46  96000]
[     47 113000]
[     33  28000]
[     19  25000]
[     49  89000]
[     31  15000]
[     30  79000]
[     48 141000]
[     32 117000]
[     37  71000]
[     18  86000]
[     42  79000]
[     27  84000]
[     40  65000]
[     57  74000]
```

```
[      26   15000]
[      26   80000]
[      29   43000]
[      33  149000]
[      39   42000]
[      54  104000]
[      36   33000]
[      46   32000]
[      40  142000]
[      37   62000]
[      29  148000]
[      37   57000]
[      35   50000]
[      42   53000]
[      35   38000]
[      41   30000]
[      40   72000]
[      26   15000]
[      31   68000]
[      35   53000]
[      35   25000]
[      30   89000]
[      41   72000]
[      28  123000]
[      46   82000]
[      22   63000]
[      45   22000]
[      30   49000]
[      34   25000]
[      40   75000]
[      32  117000]
[      23   82000]
[      26   80000]
[      48  131000]
[      59  143000]
[      35   55000]
[      34   43000]
[      39   61000]
[      27   96000]
[      60   83000]
[      24   55000]
[      58  144000]
[      53  104000]
[      35   79000]
[      36   99000]
[      57   60000]
[      37  137000]
[      33   43000]
```

```
[     41   71000]
[     52   21000]
[     52  150000]
[     37   70000]
[     26   84000]
[     26   72000]
[     26   52000]
[     41   60000]
[     31   66000]
[     37  144000]
[     38   61000]
[     31   34000]
[     42   75000]
[     46  117000]
[     36   52000]
[     38   71000]
[     49   88000]
[     57   33000]
[     48  138000]
[     47   50000]
[     33   69000]
[     37  146000]
[     20   82000]
[     40   47000]
[     35   22000]
[     20   36000]
[     45   45000]
[     26   43000]
[     58  101000]
[     40   57000]
[     38  112000]
[     37   80000]
[     49   28000]
[     36   75000]
[     41   72000]
[     35   60000]
[     43  129000]
[     41   87000]
[     38  113000]
[     58   23000]
[     26   32000]
[     32   18000]
[     41   52000]
[     31   18000]
[     35   88000]
[     48   35000]
[     27   89000]
[     35   97000]
```

```
[     42  73000]
[     21  68000]
[     41  72000]
[     33  60000]
[     39 134000]
[     28  84000]
[     46  88000]
[     24  58000]
[     31 118000]
[     50  88000]
[     20  82000]
[     32 135000]
[     20  86000]
[     35  27000]
[     29  43000]
[     21  88000]
[     35  59000]
[     45  32000]
[     60  42000]
[     35  91000]
[     35  44000]
[     18  44000]
[     42 149000]
[     45  79000]
[     40  60000]
[     24  23000]
[     33  51000]
[     42  70000]
[     55 130000]
[     50  44000]
[     48 119000]
[     19  76000]
[     41  72000]
[     40  71000]
[     27  88000]
[     36 126000]
[     35  75000]
[     35  58000]
[     34 115000]
[     35  73000]
[     60 108000]
[     25  87000]
[     27  54000]
[     21  16000]
[     37  74000]
[     35  39000]
[     54  70000]
[     47  30000]
```

```
 [     38  50000]
 [     35 147000]
 [     35  77000]
 [     41  79000]
 [     37  33000]
 [     60  46000]
 [     28  59000]
 [     23  66000]
 [     23  63000]
 [     30  17000]
 [     25  33000]
 [     59  83000]
 [     58  38000]
 [     18  82000]
 [     46  59000]
 [     27  17000]
 [     58  47000]
 [     48  30000]
 [     49  65000]
 [     50  36000]
 [     53  72000]
 [     40  57000]
 [     52 114000]
 [     59  42000]
 [     36  63000]
 [     42 104000]
 [     37  52000]
 [     48  33000]
 [     59  29000]
 [     37  79000]
 [     40  61000]
 [     49  74000]
 [     25  90000]
 [     30  15000]
 [     40  78000]
 [     24  84000]
 [     38  50000]
 [     45 131000]
 [     21  72000]
 [     35  23000]
 [     35  20000]
 [     31  89000]
 [     30  80000]
 [     47  47000]
 [     27  90000]
 [     35  72000]
 [     30 116000]
 [     39 122000]
```

```
[     29   83000]
[     41   63000]
[     48   90000]
[     38   59000]
[     32   18000]
[     39   75000]
[     26   81000]
[     39  106000]
[     22   55000]
[     36  118000]
[     60   42000]
[     28   55000]
[     51  134000]
[     49   28000]
[     36   60000]
[     56  104000]
[     27   58000]
[     24   32000]
[     34   72000]
[     28   32000]
[     50   20000]
[     33   41000]
[     29   47000]
[     22   18000]
[     30  135000]
[     47  105000]
[     46   79000]
[     48  134000]
[     47   49000]
[     49  141000]
[     32  100000]
[     38   71000]
[     19   26000]
[     37   77000]
[     47   51000]
[     40   57000]
[     36  125000]
[     20   74000]
[     31   58000]
[     41   45000]
[     42   54000]
[     28   37000]
[     39   73000]
[     28   85000]
[     38   51000]
[     47   43000]
[     37   72000]
[     49   36000]
```

```
 [     45  22000]
 [     35  72000]
 [     24  27000]
 [     26  35000]
 [     43 133000]
 [     39  77000]
 [     32  86000]]
```

[ ]:
```python
print(y_train)
```

```
[0 1 1 0 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 1 0 0 1 1 1 1 0 0 0 1 1 1 0 0
 1 0 0 0 0 1 1 1 0 0 0 0 1 1 0 1 0 0 0 1 1 0 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0
 0 0 1 0 1 0 1 1 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 1 0 0 0
 1 1 0 1 1 0 1 1 1 0 0 1 1 1 0 0 0 0 0 1 0 0 0 1 0 0 1 1 1 1 0 1 0 0 0 0 1
 0 1 0 0 0 1 0 0 0 1 1 1 1 0 0 0 0 0 1 0 1 1 0 0 0 1 0 1 0 1 1 0 1 0 0 0 0
 0 1 1 1 0 0 1 0 0 0 0 0 1 0 1 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 1 0 1 0 0 0
 1 0 0 0 0 0 0 1 0 0 0 1 1 0 1 1 0 0 0 0 1 0 1 1 1 0 0 0 0 0 0 0 1 0 0 0 0
 0 0 0 0 0 1 0 0 1 0 0 1 0 1 0 1 1 0 0 1 0 1 0 0 0 0 1 0 0 0 1 1 1 1 1 1 1
 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0]
```

[ ]:
```python
print(x_test)
```

```
[[     46  22000]
 [     59  88000]
 [     28  44000]
 [     48  96000]
 [     29  28000]
 [     30  62000]
 [     47 107000]
 [     29  83000]
 [     40  75000]
 [     42  65000]
 [     35  65000]
 [     53  34000]
 [     23  48000]
 [     20  23000]
 [     30  87000]
 [     35 108000]
 [     52  38000]
 [     46  74000]
 [     39  42000]
 [     56  60000]
 [     22  27000]
 [     29  80000]
 [     47  23000]
 [     59  76000]
 [     19  19000]
 [     51  23000]
```

```
[    42  80000]
[    37  53000]
[    55 125000]
[    19  21000]
[    46  41000]
[    19  70000]
[    36 144000]
[    28  79000]
[    40 107000]
[    35  75000]
[    37  55000]
[    38  65000]
[    26  30000]
[    18  68000]
[    48  33000]
[    24  55000]
[    18  52000]
[    47  20000]
[    20  49000]
[    44 139000]
[    29  61000]
[    31  71000]
[    48  41000]
[    34  43000]
[    25  79000]
[    53  82000]
[    38  55000]
[    25  80000]
[    37  80000]
[    27  31000]
[    60 102000]
[    26 118000]
[    31  74000]
[    31  76000]
[    41  51000]
[    35  57000]
[    52  90000]
[    28  87000]
[    28  59000]
[    27 137000]
[    42  90000]
[    47 144000]
[    39  59000]
[    33 113000]
[    52 138000]
[    19  85000]
[    27  58000]
[    23  20000]
```

```
[     47   34000]
[     35   50000]
[     56  133000]
[     54   26000]
[     35   47000]
[     37   78000]]
```

[ ]:
```
print(y_test)
```

```
[0 1 0 1 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 1 1 0 1 0 0 1 0 1 0 1 0 1 0 0
 0 0 0 1 0 0 1 0 1 0 0 1 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0 0 1 1 1 0 0 1 0 0 0
 1 0 1 1 0 1]
```

2.4 Feature Scaling

[ ]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

[ ]:
```
print(x_train)
```

```
[[-1.06675246 -0.38634438]
 [ 0.79753468 -1.22993871]
 [ 0.11069205  1.853544  ]
 [ 0.60129393 -0.90995465]
 [ 1.87685881 -1.28811763]
 [-0.57615058  1.44629156]
 [ 0.3069328  -0.53179168]
 [ 0.99377543  0.10817643]
 [-1.16487283  0.45724994]
 [-1.55735433  0.31180264]
 [ 1.0918958   0.45724994]
 [-0.18366908 -0.47361276]
 [ 0.20881242 -0.32816546]
 [ 0.3069328   0.28271318]
 [-1.16487283 -1.57901222]
 [ 0.11069205  0.25362372]
 [ 2.07309956  1.73718616]
 [ 0.40505317 -0.18271817]
 [ 1.4843773   2.11534913]
 [-0.37990983  1.21357589]
 [ 1.87685881  1.50447048]
 [ 0.11069205  0.02090805]
 [ 0.89565505 -1.31720709]
 [-1.36111358 -1.49174384]
 [-0.18366908 -0.5899706 ]
 [-0.57615058  2.31897535]
```

```
[ 0.99377543 -1.20084925]
[-0.77239133  1.06812859]
[ 2.17121993 -1.05540195]
[ 0.01257167 -0.26998655]
[-0.4780302  -1.14267033]
[ 0.11069205  0.02090805]
[ 1.68061805 -0.90995465]
[ 1.0918958  -0.90995465]
[ 0.50317355  1.21357589]
[-1.06675246 -1.46265438]
[-1.16487283 -1.54992276]
[-0.0855487   0.66087616]
[ 0.40505317 -0.47361276]
[-0.28178945 -0.26998655]
[-0.87051171  0.13726589]
[ 0.01257167  0.28271318]
[ 0.6994143  -1.28811763]
[ 1.58249768  1.09721805]
[ 0.79753468 -1.37538601]
[-1.45923396 -1.22993871]
[-0.0855487   0.13726589]
[ 0.40505317 -0.15362871]
[-0.28178945  0.02090805]
[ 1.28813655  2.20261751]
[ 0.11069205  0.74814454]
[-1.36111358  0.54451832]
[ 1.97497918  0.71905508]
[-1.26299321 -1.40447546]
[ 0.3069328  -0.32816546]
[-0.96863208  0.54451832]
[ 0.40505317  0.28271318]
[ 0.40505317  1.09721805]
[ 0.79753468  0.74814454]
[ 0.89565505  1.24266535]
[-0.4780302  -1.22993871]
[-1.85171546 -1.31720709]
[ 1.0918958   0.54451832]
[-0.67427095 -1.60810168]
[-0.77239133  0.25362372]
[ 0.99377543  2.05717021]
[-0.57615058  1.35902318]
[-0.0855487   0.02090805]
[-1.94983583  0.45724994]
[ 0.40505317  0.25362372]
[-1.06675246  0.39907102]
[ 0.20881242 -0.15362871]
[ 1.87685881  0.10817643]
[-1.16487283 -1.60810168]
```

```
[-1.16487283  0.28271318]
[-0.87051171 -0.79359682]
[-0.4780302   2.28988589]
[ 0.11069205 -0.82268628]
[ 1.58249768  0.98086021]
[-0.18366908 -1.08449141]
[ 0.79753468 -1.11358087]
[ 0.20881242  2.08625967]
[-0.0855487  -0.24089709]
[-0.87051171  2.26079643]
[-0.0855487  -0.38634438]
[-0.28178945 -0.5899706 ]
[ 0.40505317 -0.50270222]
[-0.28178945 -0.93904411]
[ 0.3069328  -1.17175979]
[ 0.20881242  0.04999751]
[-1.16487283 -1.60810168]
[-0.67427095 -0.06636033]
[-0.28178945 -0.50270222]
[-0.28178945 -1.31720709]
[-0.77239133  0.54451832]
[ 0.3069328   0.04999751]
[-0.96863208  1.53355994]
[ 0.79753468  0.3408921 ]
[-1.55735433 -0.21180763]
[ 0.6994143  -1.40447546]
[-0.77239133 -0.61906006]
[-0.37990983 -1.31720709]
[ 0.20881242  0.13726589]
[-0.57615058  1.35902318]
[-1.45923396  0.3408921 ]
[-1.16487283  0.28271318]
[ 0.99377543  1.76627562]
[ 2.07309956  2.11534913]
[-0.28178945 -0.4445233 ]
[-0.37990983 -0.79359682]
[ 0.11069205 -0.26998655]
[-1.06675246  0.74814454]
[ 2.17121993  0.36998156]
[-1.36111358 -0.4445233 ]
[ 1.97497918  2.14443859]
[ 1.4843773   0.98086021]
[-0.28178945  0.25362372]
[-0.18366908  0.83541291]
[ 1.87685881 -0.29907601]
[-0.0855487   1.94081237]
[-0.4780302  -0.79359682]
[ 0.3069328   0.02090805]
```

```
[ 1.38625693 -1.43356492]
[ 1.38625693  2.31897535]
[-0.0855487  -0.00818141]
[-1.16487283  0.39907102]
[-1.16487283  0.04999751]
[-1.16487283 -0.53179168]
[ 0.3069328  -0.29907601]
[-0.67427095 -0.12453925]
[-0.0855487   2.14443859]
[ 0.01257167 -0.26998655]
[-0.67427095 -1.05540195]
[ 0.40505317  0.13726589]
[ 0.79753468  1.35902318]
[-0.18366908 -0.53179168]
[ 0.01257167  0.02090805]
[ 1.0918958   0.51542886]
[ 1.87685881 -1.08449141]
[ 0.99377543  1.96990183]
[ 0.89565505 -0.5899706 ]
[-0.4780302  -0.03727087]
[-0.0855487   2.20261751]
[-1.75359508  0.3408921 ]
[ 0.20881242 -0.67723898]
[-0.28178945 -1.40447546]
[-1.75359508 -0.99722303]
[ 0.6994143  -0.7354179 ]
[-1.16487283 -0.79359682]
[ 1.97497918  0.89359183]
[ 0.20881242 -0.38634438]
[ 0.01257167  1.21357589]
[-0.0855487   0.28271318]
[ 1.0918958  -1.22993871]
[-0.18366908  0.13726589]
[ 0.3069328   0.04999751]
[-0.28178945 -0.29907601]
[ 0.50317355  1.7080967 ]
[ 0.3069328   0.4863394 ]
[ 0.01257167  1.24266535]
[ 1.97497918 -1.37538601]
[-1.16487283 -1.11358087]
[-0.57615058 -1.5208333 ]
[ 0.3069328  -0.53179168]
[-0.67427095 -1.5208333 ]
[-0.28178945  0.51542886]
[ 0.99377543 -1.02631249]
[-1.06675246  0.54451832]
[-0.28178945  0.777234  ]
[ 0.40505317  0.07908697]
```

```
[-1.65547471 -0.06636033]
[ 0.3069328   0.04999751]
[-0.4780302  -0.29907601]
[ 0.11069205  1.853544  ]
[-0.96863208  0.39907102]
[ 0.79753468  0.51542886]
[-1.36111358 -0.35725492]
[-0.67427095  1.38811264]
[ 1.19001618  0.51542886]
[-1.75359508  0.3408921 ]
[-0.57615058  1.88263345]
[-1.75359508  0.45724994]
[-0.28178945 -1.25902817]
[-0.87051171 -0.79359682]
[-1.65547471  0.51542886]
[-0.28178945 -0.32816546]
[ 0.6994143  -1.11358087]
[ 2.17121993 -0.82268628]
[-0.28178945  0.60269724]
[-0.28178945 -0.76450736]
[-1.94983583 -0.76450736]
[ 0.40505317  2.28988589]
[ 0.6994143   0.25362372]
[ 0.20881242 -0.29907601]
[-1.36111358 -1.37538601]
[-0.4780302  -0.56088114]
[ 0.40505317 -0.00818141]
[ 1.68061805  1.73718616]
[ 1.19001618 -0.76450736]
[ 0.99377543  1.4172021 ]
[-1.85171546  0.16635535]
[ 0.3069328   0.04999751]
[ 0.20881242  0.02090805]
[-1.06675246  0.51542886]
[-0.18366908  1.62082832]
[-0.28178945  0.13726589]
[-0.28178945 -0.35725492]
[-0.37990983  1.30084427]
[-0.28178945  0.07908697]
[ 2.17121993  1.09721805]
[-1.26299321  0.4863394 ]
[-1.06675246 -0.47361276]
[-1.65547471 -1.57901222]
[-0.0855487   0.10817643]
[-0.28178945 -0.90995465]
[ 1.58249768 -0.00818141]
[ 0.89565505 -1.17175979]
[ 0.01257167 -0.5899706 ]
```

```
[-0.28178945  2.23170697]
[-0.28178945  0.19544481]
[ 0.3069328   0.25362372]
[-0.0855487  -1.08449141]
[ 2.17121993 -0.70632844]
[-0.96863208 -0.32816546]
[-1.45923396 -0.12453925]
[-1.45923396 -0.21180763]
[-0.77239133 -1.54992276]
[-1.26299321 -1.08449141]
[ 2.07309956  0.36998156]
[ 1.97497918 -0.93904411]
[-1.94983583  0.3408921 ]
[ 0.79753468 -0.32816546]
[-1.06675246 -1.54992276]
[ 1.97497918 -0.67723898]
[ 0.99377543 -1.17175979]
[ 1.0918958  -0.15362871]
[ 1.19001618 -0.99722303]
[ 1.4843773   0.04999751]
[ 0.20881242 -0.38634438]
[ 1.38625693  1.27175481]
[ 2.07309956 -0.82268628]
[-0.18366908 -0.21180763]
[ 0.40505317  0.98086021]
[-0.0855487  -0.53179168]
[ 0.99377543 -1.08449141]
[ 2.07309956 -1.20084925]
[-0.0855487   0.25362372]
[ 0.20881242 -0.26998655]
[ 1.0918958   0.10817643]
[-1.26299321  0.57360778]
[-0.77239133 -1.60810168]
[ 0.20881242  0.22453427]
[-1.36111358  0.39907102]
[ 0.01257167 -0.5899706 ]
[ 0.6994143   1.76627562]
[-1.65547471  0.04999751]
[-0.28178945 -1.37538601]
[-0.28178945 -1.46265438]
[-0.67427095  0.54451832]
[-0.77239133  0.28271318]
[ 0.89565505 -0.67723898]
[-1.06675246  0.57360778]
[-0.28178945  0.04999751]
[-0.77239133  1.32993372]
[ 0.11069205  1.50447048]
[-0.87051171  0.36998156]
```

```
[ 0.3069328  -0.21180763]
[ 0.99377543  0.57360778]
[ 0.01257167 -0.32816546]
[-0.57615058 -1.5208333 ]
[ 0.11069205  0.13726589]
[-1.16487283  0.31180264]
[ 0.11069205  1.03903913]
[-1.55735433 -0.4445233 ]
[-0.18366908  1.38811264]
[ 2.17121993 -0.82268628]
[-0.96863208 -0.4445233 ]
[ 1.28813655  1.853544  ]
[ 1.0918958  -1.22993871]
[-0.18366908 -0.29907601]
[ 1.77873843  0.98086021]
[-1.06675246 -0.35725492]
[-1.36111358 -1.11358087]
[-0.37990983  0.04999751]
[-0.96863208 -1.11358087]
[ 1.19001618 -1.46265438]
[-0.4780302  -0.85177573]
[-0.87051171 -0.67723898]
[-1.55735433 -1.5208333 ]
[-0.77239133  1.88263345]
[ 0.89565505  1.00994967]
[ 0.79753468  0.25362372]
[ 0.99377543  1.853544  ]
[ 0.89565505 -0.61906006]
[ 1.0918958   2.05717021]
[-0.57615058  0.86450237]
[ 0.01257167  0.02090805]
[-1.85171546 -1.28811763]
[-0.0855487   0.19544481]
[ 0.89565505 -0.56088114]
[ 0.20881242 -0.38634438]
[-0.18366908  1.59173886]
[-1.75359508  0.10817643]
[-0.67427095 -0.35725492]
[ 0.3069328  -0.7354179 ]
[ 0.40505317 -0.47361276]
[-0.96863208 -0.96813357]
[ 0.11069205  0.07908697]
[-0.96863208  0.42816048]
[ 0.01257167 -0.56088114]
[ 0.89565505 -0.79359682]
[-0.0855487   0.04999751]
[ 1.0918958  -0.99722303]
[ 0.6994143  -1.40447546]
```

```
 [-0.28178945  0.04999751]
 [-1.36111358 -1.25902817]
 [-1.16487283 -1.02631249]
 [ 0.50317355  1.82445454]
 [ 0.11069205  0.19544481]
 [-0.57615058  0.45724994]]
```

[ ]:
```
print(x_test)
```

```
[[ 0.79753468 -1.40447546]
 [ 2.07309956  0.51542886]
 [-0.96863208 -0.76450736]
 [ 0.99377543  0.74814454]
 [-0.87051171 -1.22993871]
 [-0.77239133 -0.24089709]
 [ 0.89565505  1.06812859]
 [-0.87051171  0.36998156]
 [ 0.20881242  0.13726589]
 [ 0.40505317 -0.15362871]
 [-0.28178945 -0.15362871]
 [ 1.4843773  -1.05540195]
 [-1.45923396 -0.64814952]
 [-1.75359508 -1.37538601]
 [-0.77239133  0.4863394 ]
 [-0.28178945  1.09721805]
 [ 1.38625693 -0.93904411]
 [ 0.79753468  0.10817643]
 [ 0.11069205 -0.82268628]
 [ 1.77873843 -0.29907601]
 [-1.55735433 -1.25902817]
 [-0.87051171  0.28271318]
 [ 0.89565505 -1.37538601]
 [ 2.07309956  0.16635535]
 [-1.85171546 -1.49174384]
 [ 1.28813655 -1.37538601]
 [ 0.40505317  0.28271318]
 [-0.0855487  -0.50270222]
 [ 1.68061805  1.59173886]
 [-1.85171546 -1.43356492]
 [ 0.79753468 -0.85177573]
 [-1.85171546 -0.00818141]
 [-0.18366908  2.14443859]
 [-0.96863208  0.25362372]
 [ 0.20881242  1.06812859]
 [-0.28178945  0.13726589]
 [-0.0855487  -0.4445233 ]
 [ 0.01257167 -0.15362871]
 [-1.16487283 -1.17175979]
```

```
 [-1.94983583 -0.06636033]
 [ 0.99377543 -1.08449141]
 [-1.36111358 -0.4445233 ]
 [-1.94983583 -0.53179168]
 [ 0.89565505 -1.46265438]
 [-1.75359508 -0.61906006]
 [ 0.60129393  1.99899129]
 [-0.87051171 -0.26998655]
 [-0.67427095  0.02090805]
 [ 0.99377543 -0.85177573]
 [-0.37990983 -0.79359682]
 [-1.26299321  0.25362372]
 [ 1.4843773   0.3408921 ]
 [ 0.01257167 -0.4445233 ]
 [-1.26299321  0.28271318]
 [-0.0855487   0.28271318]
 [-1.06675246 -1.14267033]
 [ 2.17121993  0.92268129]
 [-1.16487283  1.38811264]
 [-0.67427095  0.10817643]
 [-0.67427095  0.16635535]
 [ 0.3069328  -0.56088114]
 [-0.28178945 -0.38634438]
 [ 1.38625693  0.57360778]
 [-0.96863208  0.4863394 ]
 [-0.96863208 -0.32816546]
 [-1.06675246  1.94081237]
 [ 0.40505317  0.57360778]
 [ 0.89565505  2.14443859]
 [ 0.11069205 -0.32816546]
 [-0.4780302   1.24266535]
 [ 1.38625693  1.96990183]
 [-1.85171546  0.42816048]
 [-1.06675246 -0.35725492]
 [-1.45923396 -1.46265438]
 [ 0.89565505 -1.05540195]
 [-0.28178945 -0.5899706 ]
 [ 1.77873843  1.82445454]
 [ 1.58249768 -1.28811763]
 [-0.28178945 -0.67723898]
 [-0.0855487   0.22453427]]
```

2.5 Training the Logistic Regression model on the Training set

[ ]:
```python
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(x_train, y_train)
```

```
[ ]: LogisticRegression(random_state=0)
```

2.6 Predicting a new result

```
[ ]: print(classifier.predict(sc.transform([[30,87000]])))
```

```
[0]
```

2.7 Predicting the Test set results

```
[ ]: y_pred = classifier.predict(x_test)
     print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.
      ↪reshape(len(y_test),1)),1))
```

```
[[0 0]
 [1 1]
 [0 0]
 [1 1]
 [0 0]
 [0 0]
 [1 1]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [1 1]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [1 1]
 [1 0]
 [0 0]
 [1 1]
 [0 0]
 [0 0]
 [0 1]
 [1 1]
 [0 0]
 [0 1]
 [1 0]
 [0 0]
 [1 1]
 [0 0]
 [0 1]
 [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 1]
[0 0]
[1 1]
[0 0]
[0 0]
[0 1]
[0 0]
[0 0]
[1 1]
[0 0]
[0 0]
[0 0]
[0 0]
[1 1]
[0 0]
[0 0]
[0 0]
[0 0]
[0 0]
[1 1]
[0 0]
[0 0]
[0 1]
[1 1]
[1 1]
[0 0]
[0 0]
[1 1]
[0 0]
[0 0]
[0 0]
[0 1]
[0 0]
[1 1]
[1 1]
[0 0]
[0 1]]
```

2.8 Making the Confusion Matrix

[ ]:
```
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
```

```
[[50  2]
 [ 9 19]]
```

[ ]:
```
0.8625
```

2.9 Visualising the Training set results

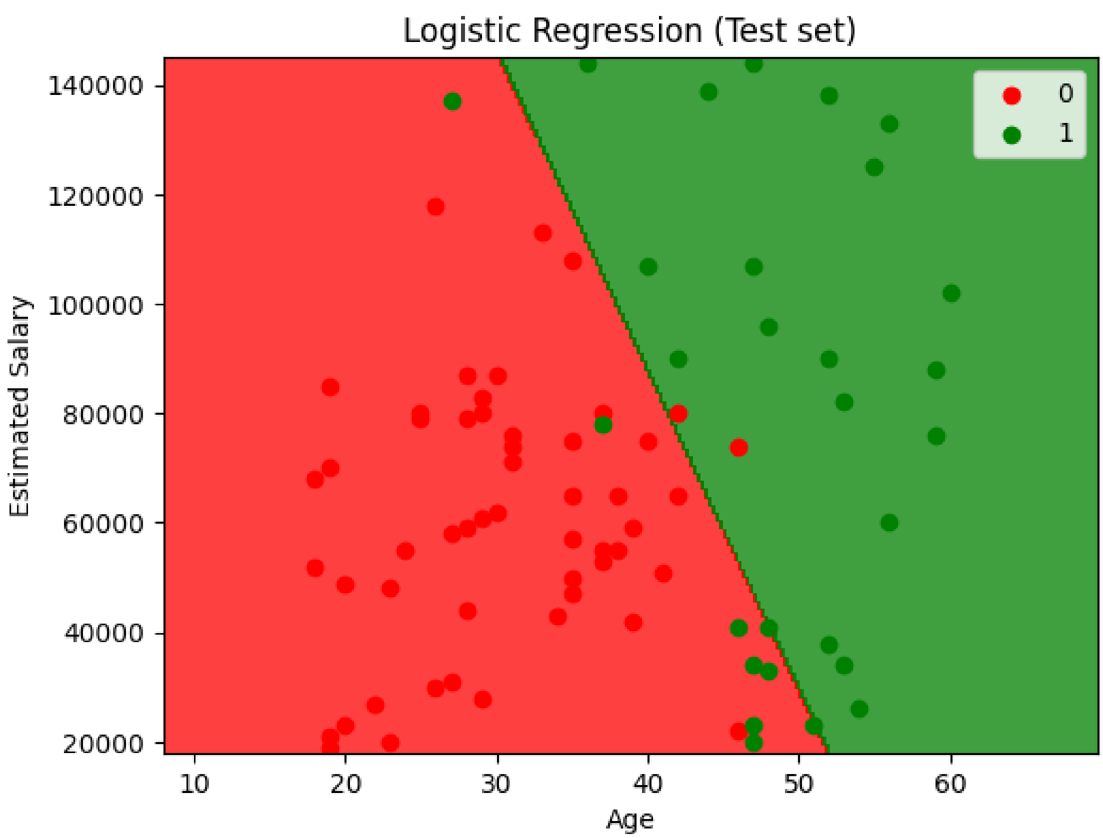
[ ]:
```
from matplotlib.colors import ListedColormap
x_set, y_set = sc.inverse_transform(x_train), y_train
x1,x2 = np.meshgrid(np.arange(start = x_set[:, 0].min() - 10, stop = x_set[:,␣
↪0].max() + 10, step = 0.25),
                     np.arange(start = x_set[:, 1].min() - 1000, stop = x_set[:
↪, 1].max() + 1000, step = 0.25))
plt.contourf(x1, x2, classifier.predict(sc.transform(np.array([x1.ravel(), x2.
↪ravel()]).T)).reshape(x1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(x1.min(), x1.max())
plt.ylim(x2.min(), x2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1], c =␣
↪ListedColormap(('red', 'green'))(i), label = j)
plt.title('Logistic Regression (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

2.10 Visualising the Test set results

[ ]:
```
from matplotlib.colors import ListedColormap
x_set, y_set = sc.inverse_transform(x_test), y_test
x1, x2 = np.meshgrid(np.arange(start = x_set[:, 0].min() - 10, stop = x_set[:,␣
↪0].max() + 10, step = 0.25),
                     np.arange(start = x_set[:, 1].min() - 1000, stop = x_set[:
↪, 1].max() + 1000, step = 0.25))
plt.contourf(x1, x2, classifier.predict(sc.transform(np.array([x1.ravel(), x2.
↪ravel()]).T)).reshape(x1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(x1.min(), x1.max())
```

```
plt.ylim(x2.min(), x2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1], c =␣
 ↪ListedColormap(('red', 'green'))(i), label = j)
plt.title('Logistic Regression (Test set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

```
<ipython-input-23-00aee6956969>:10: UserWarning: *c* argument looks like a
single numeric RGB or RGBA sequence, which should be avoided as value-mapping
will have precedence in case its length matches with *x* & *y*.  Please use the
*color* keyword-argument or provide a 2D array with a single row if you intend
to specify the same RGB or RGBA value for all points.
  plt.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1], c =
ListedColormap(('red', 'green'))(i), label = j)
```




```
[ ]:
```