logistic-regression-nrcm-1

August 26, 2023

#**Name**:S.Rajitha #**Pin**:21X05A6747 #**Branch**:Data Science #**College**:Narsimhareddy Engineering College

#**Project Title:** Prediction of "Social_Network_Ads.csv" dataset to estimate the future prediction for "Age" vs "Estimated Salary".

#**Problem Statement:** A Indian news channel "Zee24" as predicted salary estimation for finanacial year 2018 ,2019. #The organization wants to layoff the "salary" to be safe in a future by impacting huge loss. ##Task: ## As a Datascience proffesional select the particular algorithem and predict the futureistic esteemated salary.

#**Conclusion:** A model is successfully completed in Machine Learning using LinearRegression with accuracy of 0.8.

1 Logistic Regression

1.1 Importing the libraries

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

1.2 Importing the dataset

[ ]:
```
data=pd.read_csv("Social_Network_Ads.csv")
data
```

[ ]:

	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]
```

1.3 Splitting the dataset into the Training set and Test set

[ ]:
```
X=data.iloc[:,:-1].values
y=data.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.
  ↪4,random_state=0)
```

[ ]:
```
print(X_train)
```

```
[[    35  61000]
 [    21  68000]
 [    28  44000]
 [    41  87000]
 [    37  33000]
 [    27  90000]
 [    39  42000]
 [    28 123000]
 [    31 118000]
 [    25  87000]
 [    35  71000]
 [    37  70000]
 [    35  39000]
 [    47  23000]
 [    35 147000]
 [    48 138000]
 [    26  86000]
 [    25  79000]
 [    52 138000]
 [    51  23000]
 [    35  60000]
 [    33 113000]
 [    30 107000]
 [    48  33000]
 [    41  80000]
 [    48  96000]
 [    31  18000]
 [    31  71000]
 [    43 129000]
 [    59  76000]
 [    18  44000]
 [    36 118000]
 [    42  90000]
```

```
[    47  30000]
[    26  43000]
[    40  78000]
[    46  59000]
[    59  42000]
[    46  74000]
[    35  91000]
[    28  59000]
[    40  57000]
[    59 143000]
[    57  26000]
[    52  38000]
[    47 113000]
[    53 143000]
[    35  27000]
[    58 101000]
[    45  45000]
[    23  82000]
[    46  23000]
[    42  65000]
[    28  84000]
[    38  59000]
[    26  84000]
[    29  28000]
[    37  71000]
[    22  55000]
[    48  35000]
[    49  28000]
[    38  65000]
[    27  17000]
[    46  28000]
[    48 141000]
[    26  17000]
[    35  97000]
[    39  59000]
[    24  27000]
[    32  18000]
[    46  88000]
[    35  58000]
[    56  60000]
[    47  34000]
[    40  72000]
[    32 100000]
[    19  21000]
[    25  90000]
[    35  88000]
[    28  32000]
[    50  20000]
```

```
[    40  59000]
[    50  44000]
[    35  72000]
[    40 142000]
[    46  32000]
[    39  71000]
[    20  74000]
[    29  75000]
[    31  76000]
[    47  25000]
[    40  61000]
[    34 112000]
[    38  80000]
[    42  75000]
[    47  47000]
[    39  75000]
[    19  25000]
[    37  80000]
[    36  60000]
[    41  52000]
[    36 125000]
[    48  29000]
[    36 126000]
[    51 134000]
[    27  57000]
[    38  71000]
[    39  61000]
[    22  27000]
[    33  60000]
[    48  74000]
[    58  23000]
[    53  72000]
[    32 117000]
[    54  70000]
[    30  80000]
[    58  95000]
[    26  52000]
[    45  79000]
[    24  55000]
[    40  75000]
[    33  28000]
[    44 139000]
[    22  18000]
[    33  51000]
[    43 133000]
[    24  32000]
[    46  22000]
[    35  55000]
```

```
[     54 104000]
[     48 119000]
[     35  53000]
[     37 144000]
[     23  66000]
[     37 137000]
[     31  58000]
[     33  41000]
[     45  22000]
[     30  15000]
[     19  19000]
[     49  74000]
[     39 122000]
[     35  73000]
[     39  71000]
[     24  23000]
[     41  72000]
[     29  83000]
[     54  26000]
[     35  44000]
[     37  75000]
[     29  47000]
[     31  68000]
[     42  54000]
[     30 135000]
[     52 114000]
[     50  36000]
[     56 133000]
[     29  61000]
[     30  89000]
[     26  16000]
[     33  31000]
[     41  72000]
[     36  33000]
[     55 125000]
[     48 131000]
[     41  71000]
[     30  62000]
[     37  72000]
[     41  63000]
[     58  47000]
[     30 116000]
[     20  49000]
[     37  74000]
[     41  59000]
[     49  89000]
[     28  79000]
[     53  82000]
```

```
[    40  57000]
[    60  34000]
[    35 108000]
[    21  72000]
[    38  71000]
[    39 106000]
[    37  57000]
[    26  72000]
[    35  23000]
[    54 108000]
[    30  17000]
[    39 134000]
[    29  43000]
[    33  43000]
[    35  38000]
[    41  45000]
[    41  72000]
[    39 134000]
[    27 137000]
[    21  16000]
[    26  32000]
[    31  66000]
[    39  73000]
[    41  79000]
[    47  50000]
[    41  30000]
[    37  93000]
[    60  46000]
[    25  22000]
[    28  37000]
[    38  55000]
[    36  54000]
[    20  36000]
[    56 104000]
[    40  57000]
[    42 108000]
[    20  23000]
[    40  65000]
[    47  20000]
[    18  86000]
[    35  79000]
[    57  33000]
[    34  72000]
[    49  39000]
[    27  31000]
[    19  70000]
[    39  79000]
[    26  81000]
```

```
 [    25  80000]
 [    28  85000]
 [    55  39000]
 [    50  88000]
 [    49  88000]
 [    52 150000]
 [    35  65000]
 [    42  54000]
 [    34  43000]
 [    37  52000]
 [    48  30000]
 [    29  43000]
 [    36  52000]
 [    27  54000]
 [    26 118000]]
```

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

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

[ ]:
```
print(X_test)
```

```
[[    30  87000]
 [    38  50000]
 [    35  75000]
 [    30  79000]
 [    35  50000]
 [    27  20000]
 [    31  15000]
 [    36 144000]
 [    18  68000]
 [    47  43000]
 [    30  49000]
 [    28  55000]
 [    37  55000]
 [    39  77000]
 [    20  86000]
 [    32 117000]
 [    37  77000]
 [    19  85000]
 [    55 130000]
 [    35  22000]
```

```
[    35  47000]
[    47 144000]
[    41  51000]
[    47 105000]
[    23  28000]
[    49 141000]
[    28  87000]
[    29  80000]
[    37  62000]
[    32  86000]
[    21  88000]
[    37  79000]
[    57  60000]
[    37  53000]
[    24  58000]
[    18  52000]
[    22  81000]
[    34  43000]
[    31  34000]
[    49  36000]
[    27  88000]
[    41  52000]
[    27  84000]
[    35  20000]
[    43 112000]
[    27  58000]
[    37  80000]
[    52  90000]
[    26  30000]
[    49  86000]
[    57 122000]
[    34  25000]
[    35  57000]
[    34 115000]
[    59  88000]
[    45  32000]
[    29  83000]
[    26  80000]
[    49  28000]
[    23  20000]
[    32  18000]
[    60  42000]
[    19  76000]
[    36  99000]
[    19  26000]
[    60  83000]
[    24  89000]
[    27  58000]
```

```
[    40  47000]
[    42  70000]
[    32 150000]
[    35  77000]
[    22  63000]
[    45  22000]
[    27  89000]
[    18  82000]
[    42  79000]
[    40  60000]
[    53  34000]
[    47 107000]
[    58 144000]
[    59  83000]
[    24  55000]
[    26  35000]
[    58  38000]
[    42  80000]
[    40  75000]
[    59 130000]
[    46  41000]
[    41  60000]
[    42  64000]
[    37 146000]
[    23  48000]
[    25  33000]
[    24  84000]
[    27  96000]
[    23  63000]
[    48  33000]
[    48  90000]
[    42 104000]
[    44  39000]
[    32 120000]
[    38  50000]
[    32 135000]
[    52  21000]
[    53 104000]
[    39  42000]
[    38  61000]
[    36  50000]
[    36  63000]
[    35  25000]
[    35  50000]
[    42  73000]
[    47  49000]
[    59  29000]
[    49  65000]
```

```
[    45 131000]
[    31  89000]
[    46  82000]
[    47  51000]
[    26  15000]
[    60 102000]
[    38 112000]
[    40 107000]
[    42  53000]
[    35  59000]
[    48  41000]
[    48 134000]
[    38 113000]
[    29 148000]
[    26  15000]
[    60  42000]
[    24  19000]
[    42 149000]
[    46  96000]
[    28  59000]
[    39  96000]
[    28  89000]
[    41  72000]
[    45  26000]
[    33  69000]
[    20  82000]
[    31  74000]
[    42  80000]
[    35  72000]
[    33 149000]
[    40  71000]
[    51 146000]
[    46  79000]
[    35  75000]
[    38  51000]
[    36  75000]
[    37  78000]
[    38  61000]
[    60 108000]
[    20  82000]
[    57  74000]
[    42  65000]
[    26  80000]
[    46 117000]]
```

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

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

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

1.4 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)
```

```
[[-2.66990686e-01 -2.03814684e-01]
 [-1.64078656e+00  1.09774515e-03]
 [-9.53888624e-01 -7.01459153e-01]
 [ 3.21778974e-01  5.57288623e-01]
 [-7.07341328e-02 -1.02346440e+00]
 [-1.05201690e+00  6.45108235e-01]
 [ 1.25522421e-01 -7.60005562e-01]
 [-9.53888624e-01  1.61112397e+00]
 [-6.59503793e-01  1.46475795e+00]
 [-1.24827345e+00  5.57288623e-01]
 [-2.66990686e-01  8.89173575e-02]
 [-7.07341328e-02  5.96441534e-02]
 [-2.66990686e-01 -8.47825174e-01]
 [ 9.10548635e-01 -1.31619644e+00]
 [-2.66990686e-01  2.31368087e+00]
 [ 1.00867691e+00  2.05022203e+00]
 [-1.15014518e+00  5.28015419e-01]
 [-1.24827345e+00  3.23102990e-01]
 [ 1.40119002e+00  2.05022203e+00]
 [ 1.30306174e+00 -1.31619644e+00]
 [-2.66990686e-01 -2.33087888e-01]
 [-4.63247240e-01  1.31839193e+00]
 [-7.57632070e-01  1.14275271e+00]
 [ 1.00867691e+00 -1.02346440e+00]
 [ 3.21778974e-01  3.52376194e-01]
 [ 1.00867691e+00  8.20747460e-01]
 [-6.59503793e-01 -1.46256246e+00]
 [-6.59503793e-01  8.89173575e-02]
 [ 5.18035528e-01  1.78676320e+00]
 [ 2.08808796e+00  2.35283378e-01]
 [-1.93517139e+00 -7.01459153e-01]
 [-1.68862410e-01  1.46475795e+00]
 [ 4.19907251e-01  6.45108235e-01]
 [ 9.10548635e-01 -1.11128401e+00]
```

```
[-1.15014518e+00 -7.30732357e-01]
[ 2.23650697e-01  2.93829786e-01]
[ 8.12420358e-01 -2.62361092e-01]
[ 2.08808796e+00 -7.60005562e-01]
[ 8.12420358e-01  1.76736970e-01]
[-2.66990686e-01  6.74381440e-01]
[-9.53888624e-01 -2.62361092e-01]
[ 2.23650697e-01 -3.20907500e-01]
[ 2.08808796e+00  2.19658805e+00]
[ 1.89183140e+00 -1.22837683e+00]
[ 1.40119002e+00 -8.77098378e-01]
[ 9.10548635e-01  1.31839193e+00]
[ 1.49931830e+00  2.19658805e+00]
[-2.66990686e-01 -1.19910362e+00]
[ 1.98995968e+00  9.67113481e-01]
[ 7.14292081e-01 -6.72185949e-01]
[-1.44453001e+00  4.10922603e-01]
[ 8.12420358e-01 -1.31619644e+00]
[ 4.19907251e-01 -8.67218672e-02]
[-9.53888624e-01  4.69469011e-01]
[ 2.73941439e-02 -2.62361092e-01]
[-1.15014518e+00  4.69469011e-01]
[-8.55760347e-01 -1.16983042e+00]
[-7.07341328e-02  8.89173575e-02]
[-1.54265828e+00 -3.79453908e-01]
[ 1.00867691e+00 -9.64917990e-01]
[ 1.10680519e+00 -1.16983042e+00]
[ 2.73941439e-02 -8.67218672e-02]
[-1.05201690e+00 -1.49183566e+00]
[ 8.12420358e-01 -1.16983042e+00]
[ 1.00867691e+00  2.13804164e+00]
[-1.15014518e+00 -1.49183566e+00]
[-2.66990686e-01  8.50020664e-01]
[ 1.25522421e-01 -2.62361092e-01]
[-1.34640173e+00 -1.19910362e+00]
[-5.61375517e-01 -1.46256246e+00]
[ 8.12420358e-01  5.86561827e-01]
[-2.66990686e-01 -2.91634296e-01]
[ 1.79370313e+00 -2.33087888e-01]
[ 9.10548635e-01 -9.94191194e-01]
[ 2.23650697e-01  1.18190562e-01]
[-5.61375517e-01  9.37840277e-01]
[-1.83704311e+00 -1.37474285e+00]
[-1.24827345e+00  6.45108235e-01]
[-2.66990686e-01  5.86561827e-01]
[-9.53888624e-01 -1.05273760e+00]
[ 1.20493346e+00 -1.40401605e+00]
[ 2.23650697e-01 -2.62361092e-01]
```

```
[ 1.20493346e+00 -7.01459153e-01]
[-2.66990686e-01  1.18190562e-01]
[ 2.23650697e-01  2.16731485e+00]
[ 8.12420358e-01 -1.05273760e+00]
[ 1.25522421e-01  8.89173575e-02]
[-1.73891484e+00  1.76736970e-01]
[-8.55760347e-01  2.06010174e-01]
[-6.59503793e-01  2.35283378e-01]
[ 9.10548635e-01 -1.25765003e+00]
[ 2.23650697e-01 -2.03814684e-01]
[-3.65118963e-01  1.28911873e+00]
[ 2.73941439e-02  3.52376194e-01]
[ 4.19907251e-01  2.06010174e-01]
[ 9.10548635e-01 -6.13639541e-01]
[ 1.25522421e-01  2.06010174e-01]
[-1.83704311e+00 -1.25765003e+00]
[-7.07341328e-02  3.52376194e-01]
[-1.68862410e-01 -2.33087888e-01]
[ 3.21778974e-01 -4.67273521e-01]
[-1.68862410e-01  1.66967038e+00]
[ 1.00867691e+00 -1.14055721e+00]
[-1.68862410e-01  1.69894358e+00]
[ 1.30306174e+00  1.93312922e+00]
[-1.05201690e+00 -3.20907500e-01]
[ 2.73941439e-02  8.89173575e-02]
[ 1.25522421e-01 -2.03814684e-01]
[-1.54265828e+00 -1.19910362e+00]
[-4.63247240e-01 -2.33087888e-01]
[ 1.00867691e+00  1.76736970e-01]
[ 1.98995968e+00 -1.31619644e+00]
[ 1.49931830e+00  1.18190562e-01]
[-5.61375517e-01  1.43548475e+00]
[ 1.59744657e+00  5.96441534e-02]
[-7.57632070e-01  3.52376194e-01]
[ 1.98995968e+00  7.91474256e-01]
[-1.15014518e+00 -4.67273521e-01]
[ 7.14292081e-01  3.23102990e-01]
[-1.34640173e+00 -3.79453908e-01]
[ 2.23650697e-01  2.06010174e-01]
[-4.63247240e-01 -1.16983042e+00]
[ 6.16163804e-01  2.07949524e+00]
[-1.54265828e+00 -1.46256246e+00]
[-4.63247240e-01 -4.96546725e-01]
[ 5.18035528e-01  1.90385601e+00]
[-1.34640173e+00 -1.05273760e+00]
[ 8.12420358e-01 -1.34546964e+00]
[-2.66990686e-01 -3.79453908e-01]
[ 1.59744657e+00  1.05493309e+00]
```

```
[ 1.00867691e+00  1.49403115e+00]
[-2.66990686e-01 -4.38000316e-01]
[-7.07341328e-02  2.22586126e+00]
[-1.44453001e+00 -5.74486631e-02]
[-7.07341328e-02  2.02094883e+00]
[-6.59503793e-01 -2.91634296e-01]
[-4.63247240e-01 -7.89278766e-01]
[ 7.14292081e-01 -1.34546964e+00]
[-7.57632070e-01 -1.55038207e+00]
[-1.83704311e+00 -1.43328926e+00]
[ 1.10680519e+00  1.76736970e-01]
[ 1.25522421e-01  1.58185077e+00]
[-2.66990686e-01  1.47463766e-01]
[ 1.25522421e-01  8.89173575e-02]
[-1.34640173e+00 -1.31619644e+00]
[ 3.21778974e-01  1.18190562e-01]
[-8.55760347e-01  4.40195807e-01]
[ 1.59744657e+00 -1.22837683e+00]
[-2.66990686e-01 -7.01459153e-01]
[-7.07341328e-02  2.06010174e-01]
[-8.55760347e-01 -6.13639541e-01]
[-6.59503793e-01  1.09774515e-03]
[ 4.19907251e-01 -4.08727112e-01]
[-7.57632070e-01  1.96240242e+00]
[ 1.40119002e+00  1.34766513e+00]
[ 1.20493346e+00 -9.35644786e-01]
[ 1.79370313e+00  1.90385601e+00]
[-8.55760347e-01 -2.03814684e-01]
[-7.57632070e-01  6.15835031e-01]
[-1.15014518e+00 -1.52110887e+00]
[-4.63247240e-01 -1.08201081e+00]
[ 3.21778974e-01  1.18190562e-01]
[-1.68862410e-01 -1.02346440e+00]
[ 1.69557485e+00  1.66967038e+00]
[ 1.00867691e+00  1.84530960e+00]
[ 3.21778974e-01  8.89173575e-02]
[-7.57632070e-01 -1.74541479e-01]
[-7.07341328e-02  1.18190562e-01]
[ 3.21778974e-01 -1.45268275e-01]
[ 1.98995968e+00 -6.13639541e-01]
[-7.57632070e-01  1.40621154e+00]
[-1.73891484e+00 -5.55093133e-01]
[-7.07341328e-02  1.76736970e-01]
[ 3.21778974e-01 -2.62361092e-01]
[ 1.10680519e+00  6.15835031e-01]
[-9.53888624e-01  3.23102990e-01]
[ 1.49931830e+00  4.10922603e-01]
[ 2.23650697e-01 -3.20907500e-01]
```

```
[ 2.18621623e+00 -9.94191194e-01]
[-2.66990686e-01  1.17202591e+00]
[-1.64078656e+00  1.18190562e-01]
[ 2.73941439e-02  8.89173575e-02]
[ 1.25522421e-01  1.11347950e+00]
[-7.07341328e-02 -3.20907500e-01]
[-1.15014518e+00  1.18190562e-01]
[-2.66990686e-01 -1.31619644e+00]
[ 1.59744657e+00  1.17202591e+00]
[-7.57632070e-01 -1.49183566e+00]
[ 1.25522421e-01  1.93312922e+00]
[-8.55760347e-01 -7.30732357e-01]
[-4.63247240e-01 -7.30732357e-01]
[-2.66990686e-01 -8.77098378e-01]
[ 3.21778974e-01 -6.72185949e-01]
[ 3.21778974e-01  1.18190562e-01]
[ 1.25522421e-01  1.93312922e+00]
[-1.05201690e+00  2.02094883e+00]
[-1.64078656e+00 -1.52110887e+00]
[-1.15014518e+00 -1.05273760e+00]
[-6.59503793e-01 -5.74486631e-02]
[ 1.25522421e-01  1.47463766e-01]
[ 3.21778974e-01  3.23102990e-01]
[ 9.10548635e-01 -5.25819929e-01]
[ 3.21778974e-01 -1.11128401e+00]
[-7.07341328e-02  7.32927848e-01]
[ 2.18621623e+00 -6.42912745e-01]
[-1.24827345e+00 -1.34546964e+00]
[-9.53888624e-01 -9.06371582e-01]
[ 2.73941439e-02 -3.79453908e-01]
[-1.68862410e-01 -4.08727112e-01]
[-1.73891484e+00 -9.35644786e-01]
[ 1.79370313e+00  1.05493309e+00]
[ 2.23650697e-01 -3.20907500e-01]
[ 4.19907251e-01  1.17202591e+00]
[-1.73891484e+00 -1.31619644e+00]
[ 2.23650697e-01 -8.67218672e-02]
[ 9.10548635e-01 -1.40401605e+00]
[-1.93517139e+00  5.28015419e-01]
[-2.66990686e-01  3.23102990e-01]
[ 1.89183140e+00 -1.02346440e+00]
[-3.65118963e-01  1.18190562e-01]
[ 1.10680519e+00 -8.47825174e-01]
[-1.05201690e+00 -1.08201081e+00]
[-1.83704311e+00  5.96441534e-02]
[ 1.25522421e-01  3.23102990e-01]
[-1.15014518e+00  3.81649399e-01]
[-1.24827345e+00  3.52376194e-01]
```

```
 [-9.53888624e-01  4.98742215e-01]
 [ 1.69557485e+00 -8.47825174e-01]
 [ 1.20493346e+00  5.86561827e-01]
 [ 1.10680519e+00  5.86561827e-01]
 [ 1.40119002e+00  2.40150048e+00]
 [-2.66990686e-01 -8.67218672e-02]
 [ 4.19907251e-01 -4.08727112e-01]
 [-3.65118963e-01 -7.30732357e-01]
 [-7.07341328e-02 -4.67273521e-01]
 [ 1.00867691e+00 -1.11128401e+00]
 [-8.55760347e-01 -7.30732357e-01]
 [-1.68862410e-01 -4.67273521e-01]
 [-1.05201690e+00 -4.08727112e-01]
 [-1.15014518e+00  1.46475795e+00]]
```

[ ]:
```
print(X_test)
```

```
[[-7.57632070e-01  5.57288623e-01]
 [ 2.73941439e-02 -5.25819929e-01]
 [-2.66990686e-01  2.06010174e-01]
 [-7.57632070e-01  3.23102990e-01]
 [-2.66990686e-01 -5.25819929e-01]
 [-1.05201690e+00 -1.40401605e+00]
 [-6.59503793e-01 -1.55038207e+00]
 [-1.68862410e-01  2.22586126e+00]
 [-1.93517139e+00  1.09774515e-03]
 [ 9.10548635e-01 -7.30732357e-01]
 [-7.57632070e-01 -5.55093133e-01]
 [-9.53888624e-01 -3.79453908e-01]
 [-7.07341328e-02 -3.79453908e-01]
 [ 1.25522421e-01  2.64556582e-01]
 [-1.73891484e+00  5.28015419e-01]
 [-5.61375517e-01  1.43548475e+00]
 [-7.07341328e-02  2.64556582e-01]
 [-1.83704311e+00  4.98742215e-01]
 [ 1.69557485e+00  1.81603640e+00]
 [-2.66990686e-01 -1.34546964e+00]
 [-2.66990686e-01 -6.13639541e-01]
 [ 9.10548635e-01  2.22586126e+00]
 [ 3.21778974e-01 -4.96546725e-01]
 [ 9.10548635e-01  1.08420630e+00]
 [-1.44453001e+00 -1.16983042e+00]
 [ 1.10680519e+00  2.13804164e+00]
 [-9.53888624e-01  5.57288623e-01]
 [-8.55760347e-01  3.52376194e-01]
 [-7.07341328e-02 -1.74541479e-01]
 [-5.61375517e-01  5.28015419e-01]
 [-1.64078656e+00  5.86561827e-01]
```

```
[-7.07341328e-02  3.23102990e-01]
[ 1.89183140e+00 -2.33087888e-01]
[-7.07341328e-02 -4.38000316e-01]
[-1.34640173e+00 -2.91634296e-01]
[-1.93517139e+00 -4.67273521e-01]
[-1.54265828e+00  3.81649399e-01]
[-3.65118963e-01 -7.30732357e-01]
[-6.59503793e-01 -9.94191194e-01]
[ 1.10680519e+00 -9.35644786e-01]
[-1.05201690e+00  5.86561827e-01]
[ 3.21778974e-01 -4.67273521e-01]
[-1.05201690e+00  4.69469011e-01]
[-2.66990686e-01 -1.40401605e+00]
[ 5.18035528e-01  1.28911873e+00]
[-1.05201690e+00 -2.91634296e-01]
[-7.07341328e-02  3.52376194e-01]
[ 1.40119002e+00  6.45108235e-01]
[-1.15014518e+00 -1.11128401e+00]
[ 1.10680519e+00  5.28015419e-01]
[ 1.89183140e+00  1.58185077e+00]
[-3.65118963e-01 -1.25765003e+00]
[-2.66990686e-01 -3.20907500e-01]
[-3.65118963e-01  1.37693834e+00]
[ 2.08808796e+00  5.86561827e-01]
[ 7.14292081e-01 -1.05273760e+00]
[-8.55760347e-01  4.40195807e-01]
[-1.15014518e+00  3.52376194e-01]
[ 1.10680519e+00 -1.16983042e+00]
[-1.44453001e+00 -1.40401605e+00]
[-5.61375517e-01 -1.46256246e+00]
[ 2.18621623e+00 -7.60005562e-01]
[-1.83704311e+00  2.35283378e-01]
[-1.68862410e-01  9.08567072e-01]
[-1.83704311e+00 -1.22837683e+00]
[ 2.18621623e+00  4.40195807e-01]
[-1.34640173e+00  6.15835031e-01]
[-1.05201690e+00 -2.91634296e-01]
[ 2.23650697e-01 -6.13639541e-01]
[ 4.19907251e-01  5.96441534e-02]
[-5.61375517e-01  2.40150048e+00]
[-2.66990686e-01  2.64556582e-01]
[-1.54265828e+00 -1.45268275e-01]
[ 7.14292081e-01 -1.34546964e+00]
[-1.05201690e+00  6.15835031e-01]
[-1.93517139e+00  4.10922603e-01]
[ 4.19907251e-01  3.23102990e-01]
[ 2.23650697e-01 -2.33087888e-01]
[ 1.49931830e+00 -9.94191194e-01]
```

```
[ 9.10548635e-01  1.14275271e+00]
[ 1.98995968e+00  2.22586126e+00]
[ 2.08808796e+00  4.40195807e-01]
[-1.34640173e+00 -3.79453908e-01]
[-1.15014518e+00 -9.64917990e-01]
[ 1.98995968e+00 -8.77098378e-01]
[ 4.19907251e-01  3.52376194e-01]
[ 2.23650697e-01  2.06010174e-01]
[ 2.08808796e+00  1.81603640e+00]
[ 8.12420358e-01 -7.89278766e-01]
[ 3.21778974e-01 -2.33087888e-01]
[ 4.19907251e-01 -1.15995071e-01]
[-7.07341328e-02  2.28440767e+00]
[-1.44453001e+00 -5.84366337e-01]
[-1.24827345e+00 -1.02346440e+00]
[-1.34640173e+00  4.69469011e-01]
[-1.05201690e+00  8.20747460e-01]
[-1.44453001e+00 -1.45268275e-01]
[ 1.00867691e+00 -1.02346440e+00]
[ 1.00867691e+00  6.45108235e-01]
[ 4.19907251e-01  1.05493309e+00]
[ 6.16163804e-01 -8.47825174e-01]
[-5.61375517e-01  1.52330436e+00]
[ 2.73941439e-02 -5.25819929e-01]
[-5.61375517e-01  1.96240242e+00]
[ 1.40119002e+00 -1.37474285e+00]
[ 1.49931830e+00  1.05493309e+00]
[ 1.25522421e-01 -7.60005562e-01]
[ 2.73941439e-02 -2.03814684e-01]
[-1.68862410e-01 -5.25819929e-01]
[-1.68862410e-01 -1.45268275e-01]
[-2.66990686e-01 -1.25765003e+00]
[-2.66990686e-01 -5.25819929e-01]
[ 4.19907251e-01  1.47463766e-01]
[ 9.10548635e-01 -5.55093133e-01]
[ 2.08808796e+00 -1.14055721e+00]
[ 1.10680519e+00 -8.67218672e-02]
[ 7.14292081e-01  1.84530960e+00]
[-6.59503793e-01  6.15835031e-01]
[ 8.12420358e-01  4.10922603e-01]
[ 9.10548635e-01 -4.96546725e-01]
[-1.15014518e+00 -1.55038207e+00]
[ 2.18621623e+00  9.96386685e-01]
[ 2.73941439e-02  1.28911873e+00]
[ 2.23650697e-01  1.14275271e+00]
[ 4.19907251e-01 -4.38000316e-01]
[-2.66990686e-01 -2.62361092e-01]
[ 1.00867691e+00 -7.89278766e-01]
```

```
 [ 1.00867691e+00  1.93312922e+00]
 [ 2.73941439e-02  1.31839193e+00]
 [-8.55760347e-01  2.34295407e+00]
 [-1.15014518e+00 -1.55038207e+00]
 [ 2.18621623e+00 -7.60005562e-01]
 [-1.34640173e+00 -1.43328926e+00]
 [ 4.19907251e-01  2.37222728e+00]
 [ 8.12420358e-01  8.20747460e-01]
 [-9.53888624e-01 -2.62361092e-01]
 [ 1.25522421e-01  8.20747460e-01]
 [-9.53888624e-01  6.15835031e-01]
 [ 3.21778974e-01  1.18190562e-01]
 [ 7.14292081e-01 -1.22837683e+00]
 [-4.63247240e-01  3.03709493e-02]
 [-1.73891484e+00  4.10922603e-01]
 [-6.59503793e-01  1.76736970e-01]
 [ 4.19907251e-01  3.52376194e-01]
 [-2.66990686e-01  1.18190562e-01]
 [-4.63247240e-01  2.37222728e+00]
 [ 2.23650697e-01  8.89173575e-02]
 [ 1.30306174e+00  2.28440767e+00]
 [ 8.12420358e-01  3.23102990e-01]
 [-2.66990686e-01  2.06010174e-01]
 [ 2.73941439e-02 -4.96546725e-01]
 [-1.68862410e-01  2.06010174e-01]
 [-7.07341328e-02  2.93829786e-01]
 [ 2.73941439e-02 -2.03814684e-01]
 [ 2.18621623e+00  1.17202591e+00]
 [-1.73891484e+00  4.10922603e-01]
 [ 1.89183140e+00  1.76736970e-01]
 [ 4.19907251e-01 -8.67218672e-02]
 [-1.15014518e+00  3.52376194e-01]
 [ 8.12420358e-01  1.43548475e+00]]
```

1.5 Training the Logistic Regression model on the Training set

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

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

1.6 Predicting a new result

[ ]:
```
print(classifier.predict(sc.transform([[59,  42000]])))
```

```
[1]
```

2 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]
 [0 0]
 [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 0]
 [0 0]
 [1 1]
 [0 0]
 [0 0]
 [1 1]
 [0 0]
 [1 1]
 [0 0]
 [1 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]
 [1 1]
 [0 0]
```

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

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

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

2.1 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)
```

```
[[93  8]
 [15 44]]
```

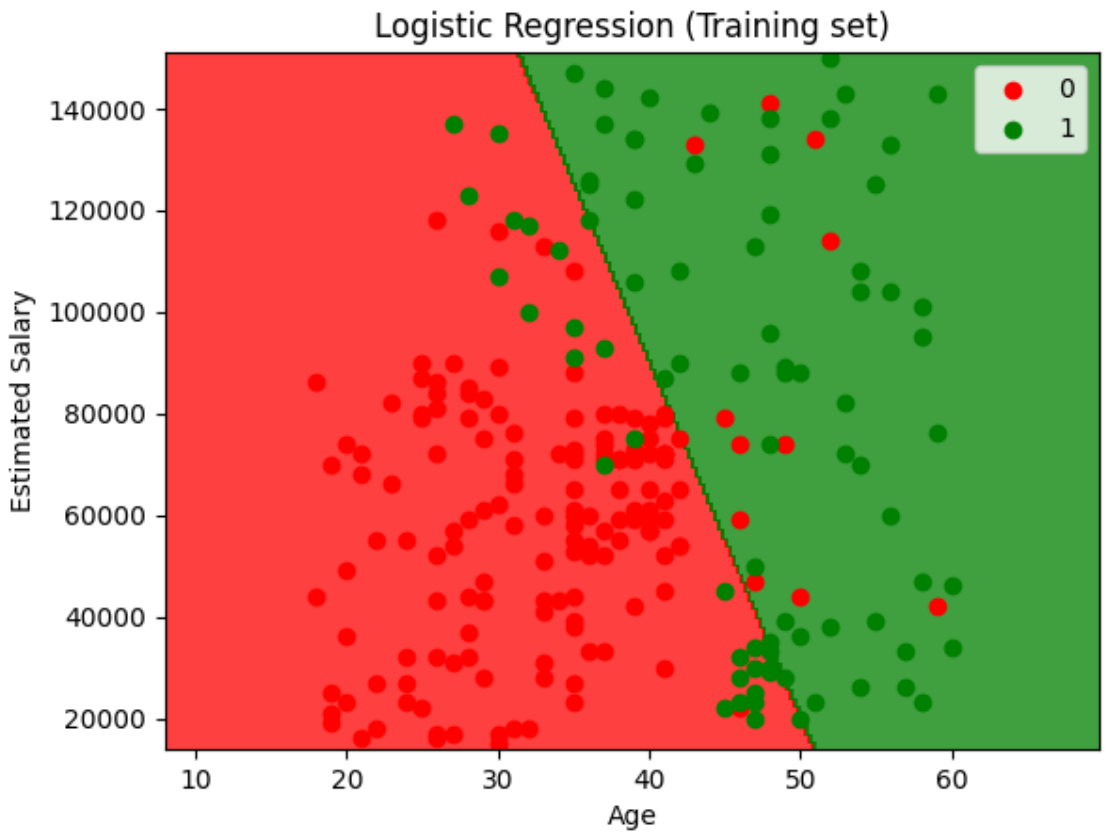
[ ]:
```
0.85625
```

2.2 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()
```

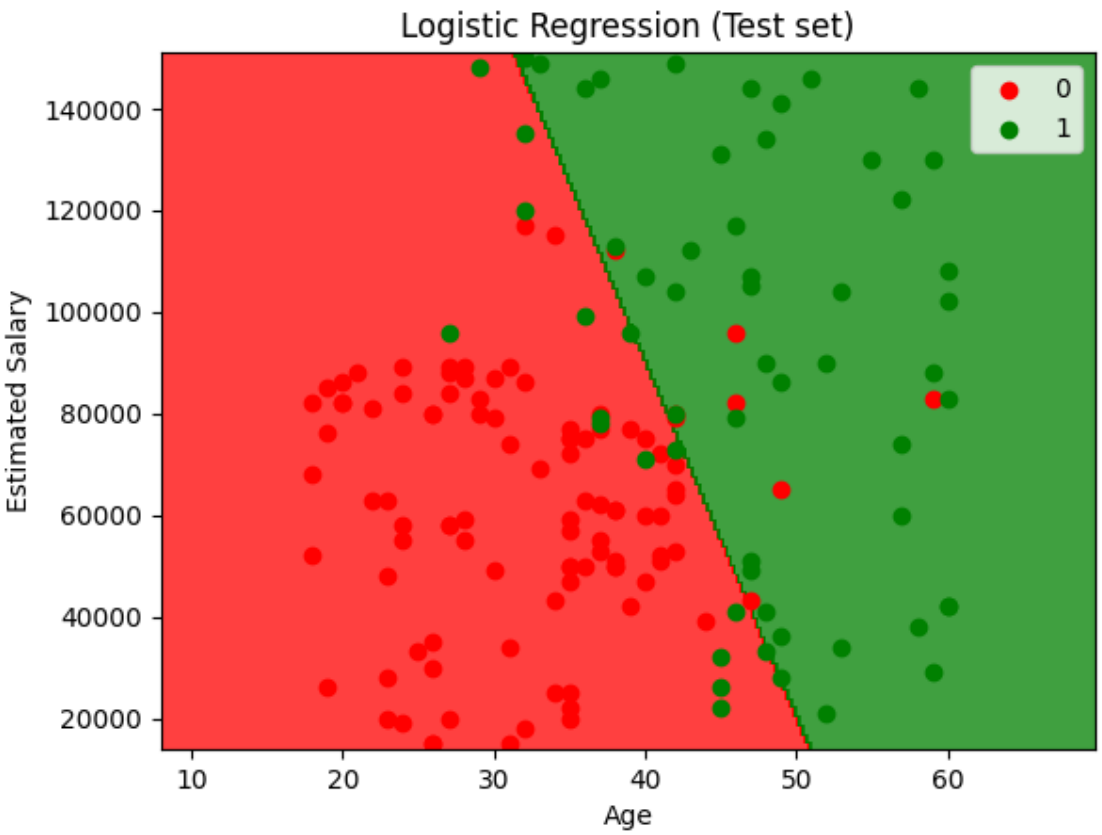
```
<ipython-input-38-3277c112bab0>:10: UserWarning: *c* argument looks like a
single numeric RGB or RGBA sequence, which should be avoided as value-mapping
will have precedence in case its length matches with *x* & *y*.  Please use the
*color* keyword-argument or provide a 2D array with a single row 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)
```



3 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-39-53d83417cfe6>: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)
```


Logistic Regression (Test set)
Estimated Salary
Age
0
1
140000
120000
100000
80000
60000
40000
20000
10
20
30
40
50
60

[ ]: