MACHINE LEARNING

# LAB 4
# AML

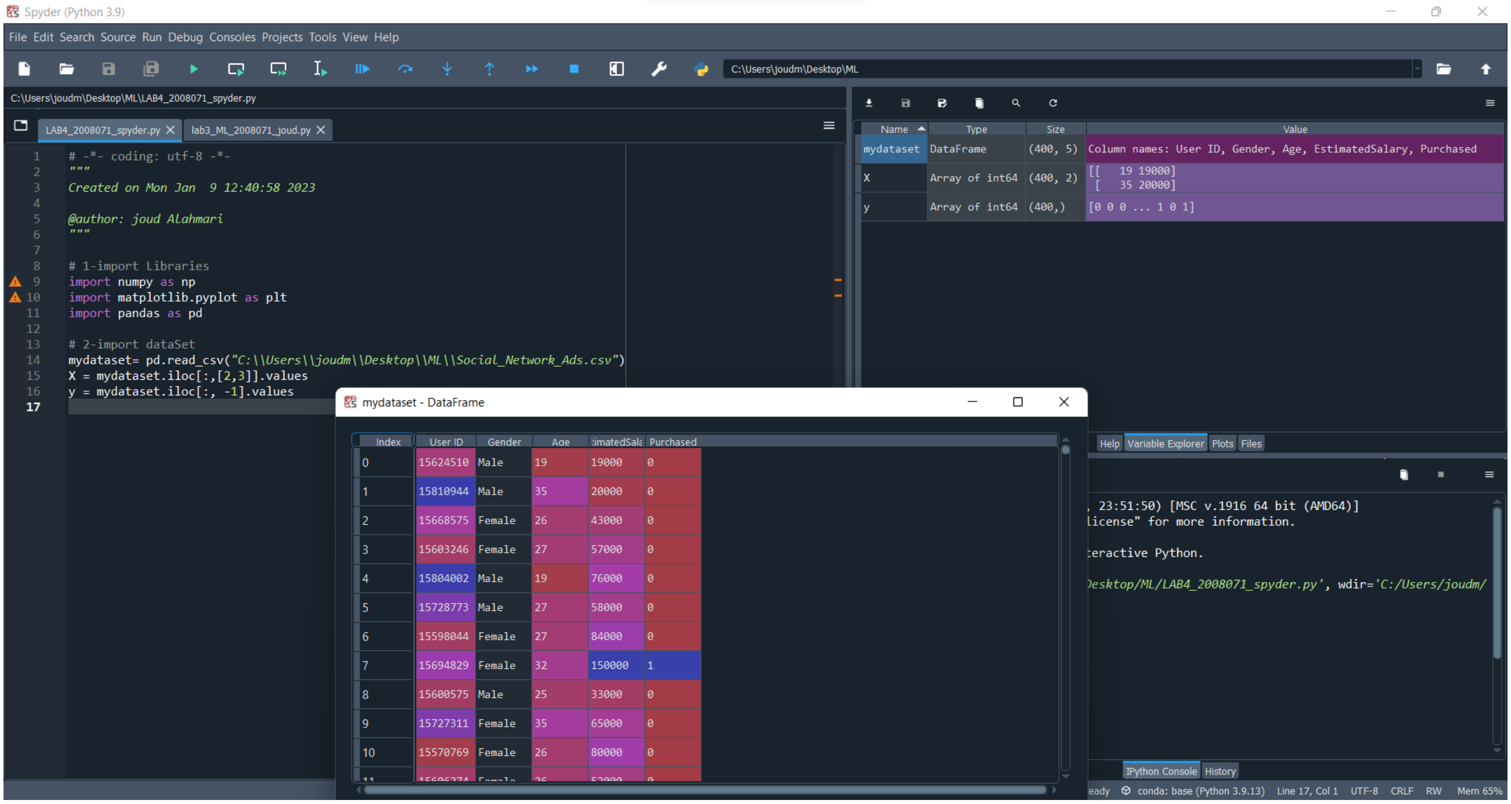
Student Name: Joud Alahmari

Student ID: 2008071

Import data set

And do the splitting of data set and training set

Note:20% for testset,80% for training set


Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\joudm\Desktop\ML
C:\Users\joudm\Desktop\ML\LAB4_2008071_spyder.py
LAB4_2008071_spyder.py
lab3_ML_2008071_joud.py
Help Variable Explorer Plots Files
IPython Console History
eady conda: base (Python 3.9.13) Line 17, Col 1 UTF-8 CRLF RW Mem 65%

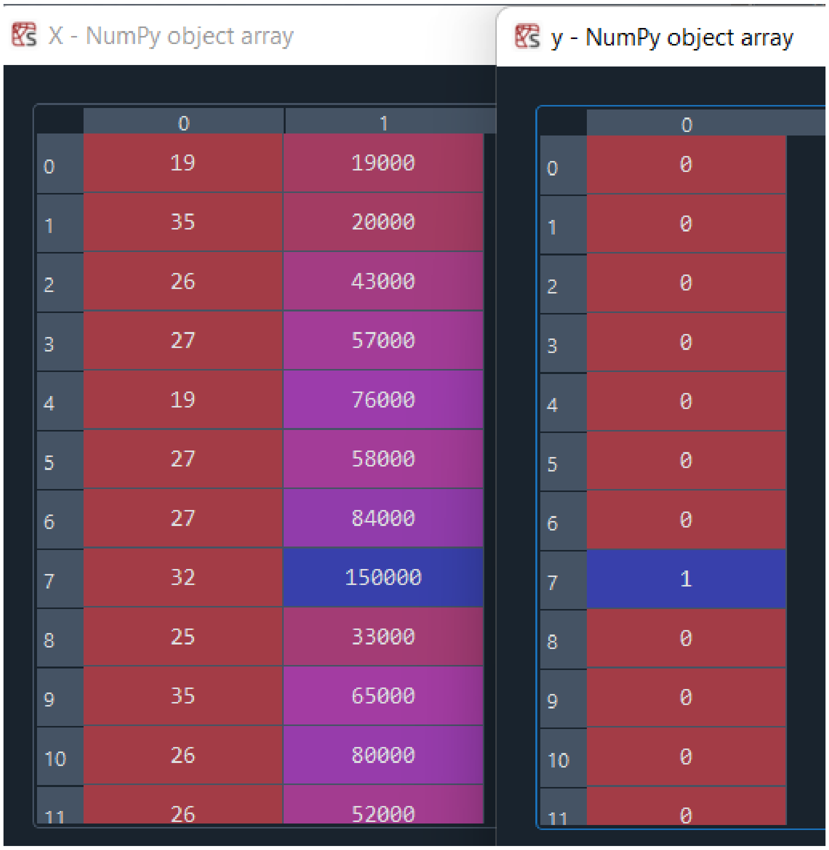
```python
# -*- coding: utf-8 -*-
"""
Created on Mon Jan  9 12:40:58 2023

@author: joud Alahmari
"""

# 1-import Libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# 2-import dataSet
mydataset= pd.read_csv("C:\\Users\\joudm\\Desktop\\ML\\Social_Network_Ads.csv")
X = mydataset.iloc[:,[2,3]].values
y = mydataset.iloc[:, -1].values
```

| Name | Type | Size | Value |
|---|---|---|---|
| mydataset | DataFrame | (400, 5) | Column names: User ID, Gender, Age, EstimatedSalary, Purchased |
| X | Array of int64 | (400, 2) | [[   19 19000]<br> [   35 20000] |
| y | Array of int64 | (400,) | [0 0 0 ... 1 0 1] |

mydataset - DataFrame

| Index | User ID | Gender | Age | imatedSal: | Purchased |
|---|---|---|---|---|---|
| 0 | 15624510 | Male | 19 | 19000 | 0 |
| 1 | 15810944 | Male | 35 | 20000 | 0 |
| 2 | 15668575 | Female | 26 | 43000 | 0 |
| 3 | 15603246 | Female | 27 | 57000 | 0 |
| 4 | 15804002 | Male | 19 | 76000 | 0 |
| 5 | 15728773 | Male | 27 | 58000 | 0 |
| 6 | 15598044 | Female | 27 | 84000 | 0 |
| 7 | 15694829 | Female | 32 | 150000 | 1 |
| 8 | 15600575 | Male | 25 | 33000 | 0 |
| 9 | 15727311 | Female | 35 | 65000 | 0 |
| 10 | 15570769 | Female | 26 | 80000 | 0 |

```
, 23:51:50) [MSC v.1916 64 bit (AMD64)]
license" for more information.

teractive Python.

Desktop/ML/LAB4_2008071_spyder.py', wdir='C:/Users/joudm/
```



X - NumPy object array

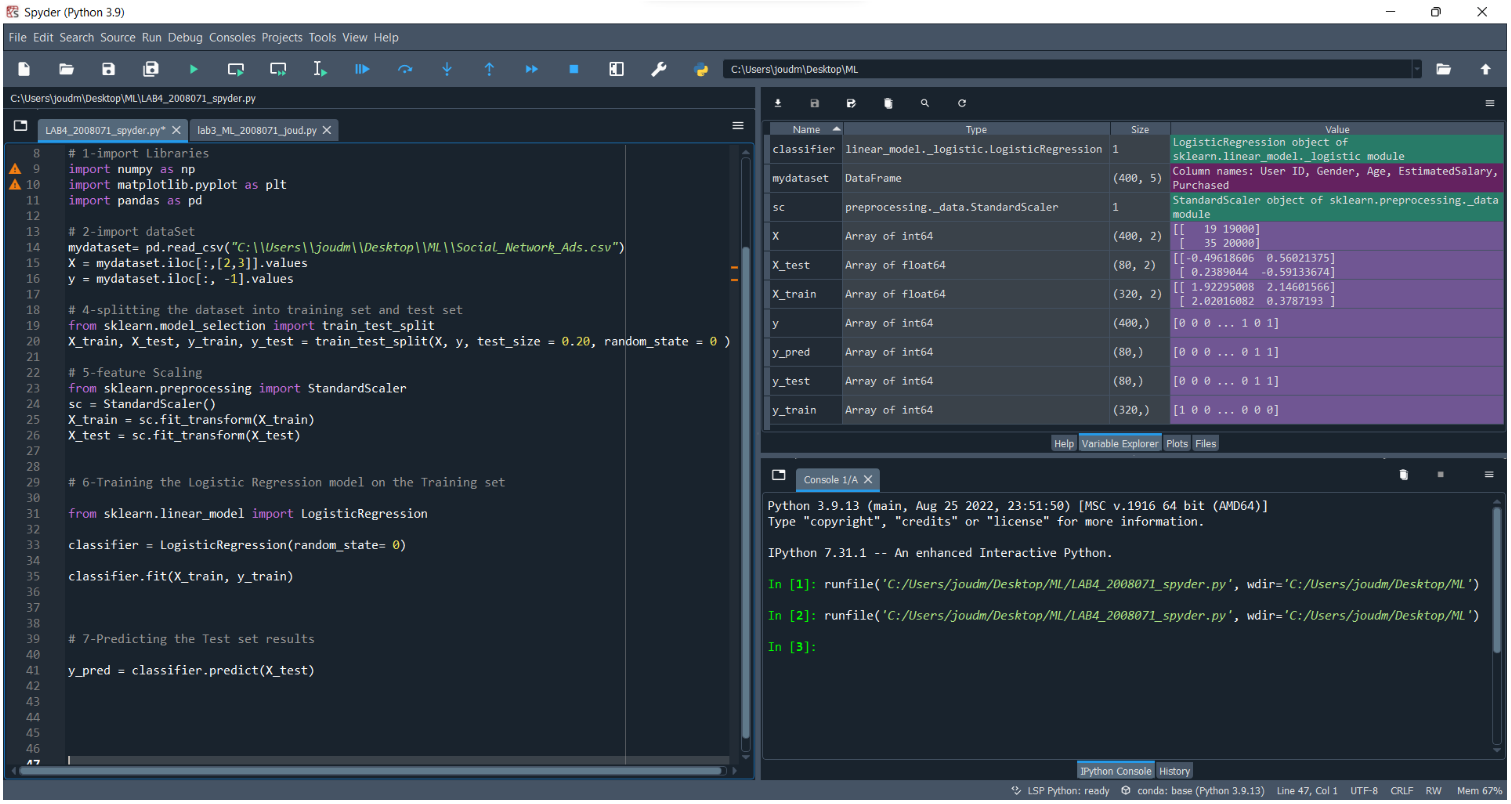
|  | 0 | 1 |
|---|---|---|
| 0 | 19 | 19000 |
| 1 | 35 | 20000 |
| 2 | 26 | 43000 |
| 3 | 27 | 57000 |
| 4 | 19 | 76000 |
| 5 | 27 | 58000 |
| 6 | 27 | 84000 |
| 7 | 32 | 150000 |
| 8 | 25 | 33000 |
| 9 | 35 | 65000 |
| 10 | 26 | 80000 |
| 11 | 26 | 52000 |

y - NumPy object array

|  | 0 |
|---|---|
| 0 | 0 |
| 1 | 0 |
| 2 | 0 |
| 3 | 0 |
| 4 | 0 |
| 5 | 0 |
| 6 | 0 |
| 7 | 1 |
| 8 | 0 |
| 9 | 0 |
| 10 | 0 |
| 11 | 0 |

# Training the Logistic Regression model on the Trainingset
# And
# predict the results

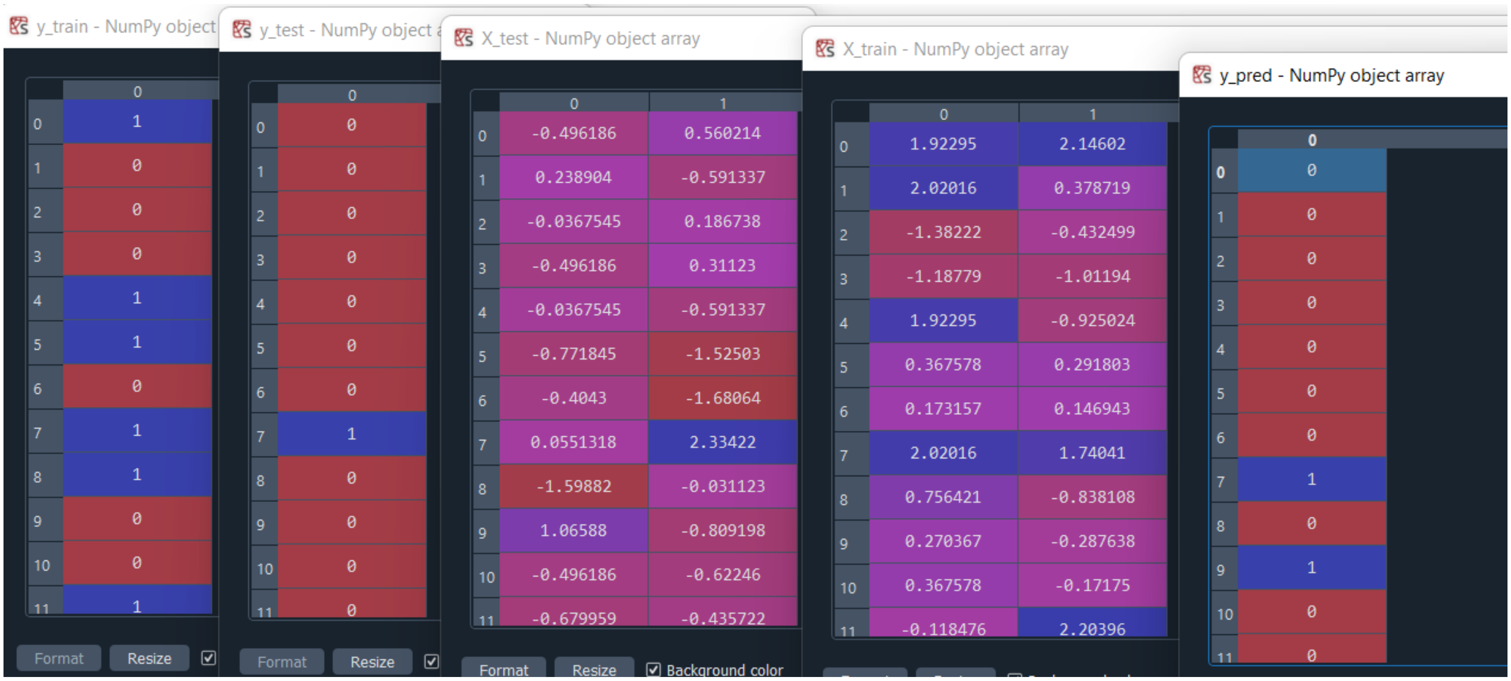
MACHINE LEARNING



```python
# 1-import Libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# 2-import dataSet
mydataset= pd.read_csv("C:\\Users\\joudm\\Desktop\\ML\\Social_Network_Ads.csv")
X = mydataset.iloc[:,[2,3]].values
y = mydataset.iloc[:, -1].values

# 4-splitting the dataset into training set and test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0 )

# 5-feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.fit_transform(X_test)


# 6-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)



# 7-Predicting the Test set results

y_pred = classifier.predict(X_test)
```

| Name | Type | Size | Value |
|---|---|---|---|
| classifier | linear_model._logistic.LogisticRegression | 1 | LogisticRegression object of sklearn.linear_model._logistic module |
| mydataset | DataFrame | (400, 5) | Column names: User ID, Gender, Age, EstimatedSalary, Purchased |
| sc | preprocessing._data.StandardScaler | 1 | StandardScaler object of sklearn.preprocessing._data module |
| X | Array of int64 | (400, 2) | [[   19 19000]<br>[   35 20000]] |
| X_test | Array of float64 | (80, 2) | [[-0.49618606  0.56021375]<br>[ 0.2389044  -0.59133674]] |
| X_train | Array of float64 | (320, 2) | [[ 1.92295008  2.14601566]<br>[ 2.02016082  0.3787193 ]] |
| y | Array of int64 | (400,) | [0 0 0 ... 1 0 1] |
| y_pred | Array of int64 | (80,) | [0 0 0 ... 0 1 1] |
| y_test | Array of int64 | (80,) | [0 0 0 ... 0 1 1] |
| y_train | Array of int64 | (320,) | [1 0 0 ... 0 0 0] |

```
Python 3.9.13 (main, Aug 25 2022, 23:51:50) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.31.1 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/joudm/Desktop/ML/LAB4_2008071_spyder.py', wdir='C:/Users/joudm/Desktop/ML')

In [2]: runfile('C:/Users/joudm/Desktop/ML/LAB4_2008071_spyder.py', wdir='C:/Users/joudm/Desktop/ML')

In [3]:
```



y_train - NumPy object

|  | 0 |
|---|---|
| 0 | 1 |
| 1 | 0 |
| 2 | 0 |
| 3 | 0 |
| 4 | 1 |
| 5 | 1 |
| 6 | 0 |
| 7 | 1 |
| 8 | 1 |
| 9 | 0 |
| 10 | 0 |
| 11 | 1 |

y_test - NumPy object a

|  | 0 |
|---|---|
| 0 | 0 |
| 1 | 0 |
| 2 | 0 |
| 3 | 0 |
| 4 | 0 |
| 5 | 0 |
| 6 | 0 |
| 7 | 1 |
| 8 | 0 |
| 9 | 0 |
| 10 | 0 |
| 11 | 0 |

X_test - NumPy object array

|  | 0 | 1 |
|---|---|---|
| 0 | -0.496186 | 0.560214 |
| 1 | 0.238904 | -0.591337 |
| 2 | -0.0367545 | 0.186738 |
| 3 | -0.496186 | 0.31123 |
| 4 | -0.0367545 | -0.591337 |
| 5 | -0.771845 | -1.52503 |
| 6 | -0.4043 | -1.68064 |
| 7 | 0.0551318 | 2.33422 |
| 8 | -1.59882 | -0.031123 |
| 9 | 1.06588 | -0.809198 |
| 10 | -0.496186 | -0.62246 |
| 11 | -0.679959 | -0.435722 |

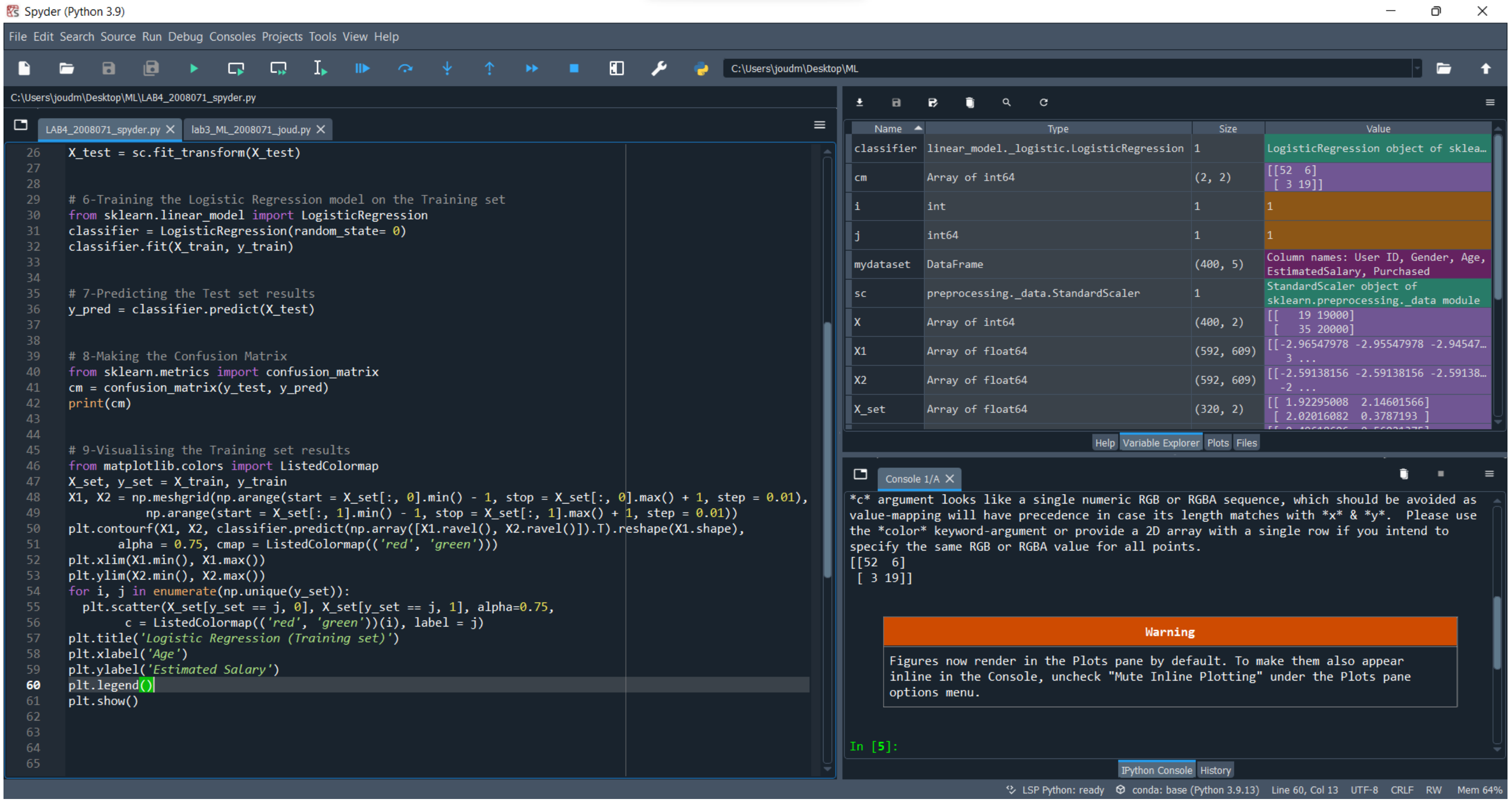
X_train - NumPy object array

|  | 0 | 1 |
|---|---|---|
| 0 | 1.92295 | 2.14602 |
| 1 | 2.02016 | 0.378719 |
| 2 | -1.38222 | -0.432499 |
| 3 | -1.18779 | -1.01194 |
| 4 | 1.92295 | -0.925024 |
| 5 | 0.367578 | 0.291803 |
| 6 | 0.173157 | 0.146943 |
| 7 | 2.02016 | 1.74041 |
| 8 | 0.756421 | -0.838108 |
| 9 | 0.270367 | -0.287638 |
| 10 | 0.367578 | -0.17175 |
| 11 | -0.118476 | 2.20396 |

y_pred - NumPy object array

|  | 0 |
|---|---|
| 0 | 0 |
| 1 | 0 |
| 2 | 0 |
| 3 | 0 |
| 4 | 0 |
| 5 | 0 |
| 6 | 0 |
| 7 | 1 |
| 8 | 0 |
| 9 | 1 |
| 10 | 0 |
| 11 | 0 |

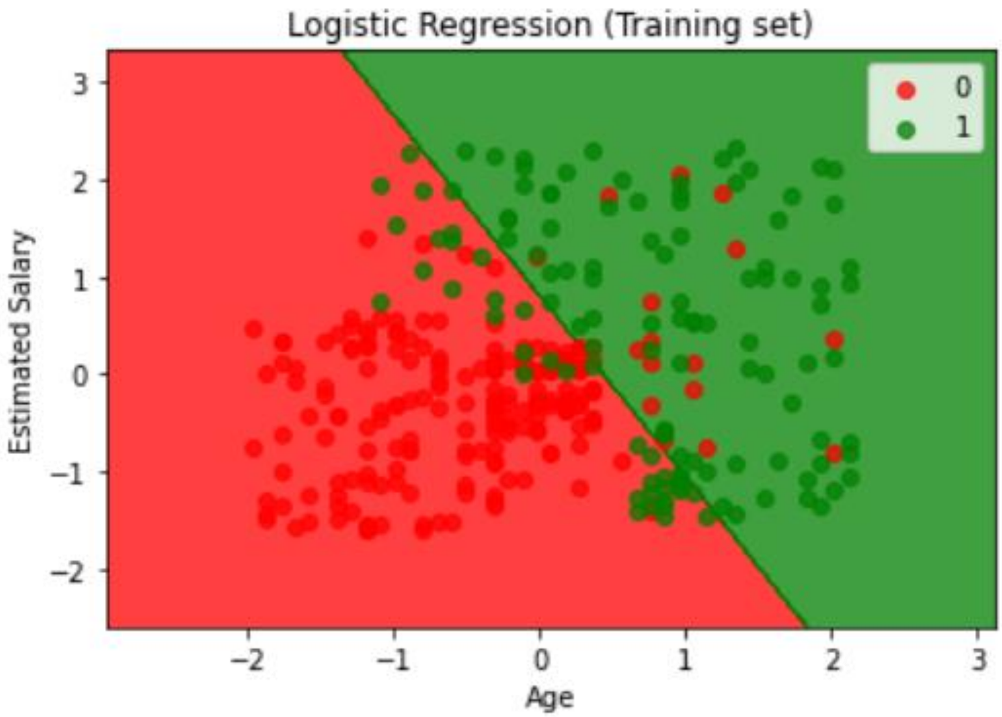
# Visualising the Training set results



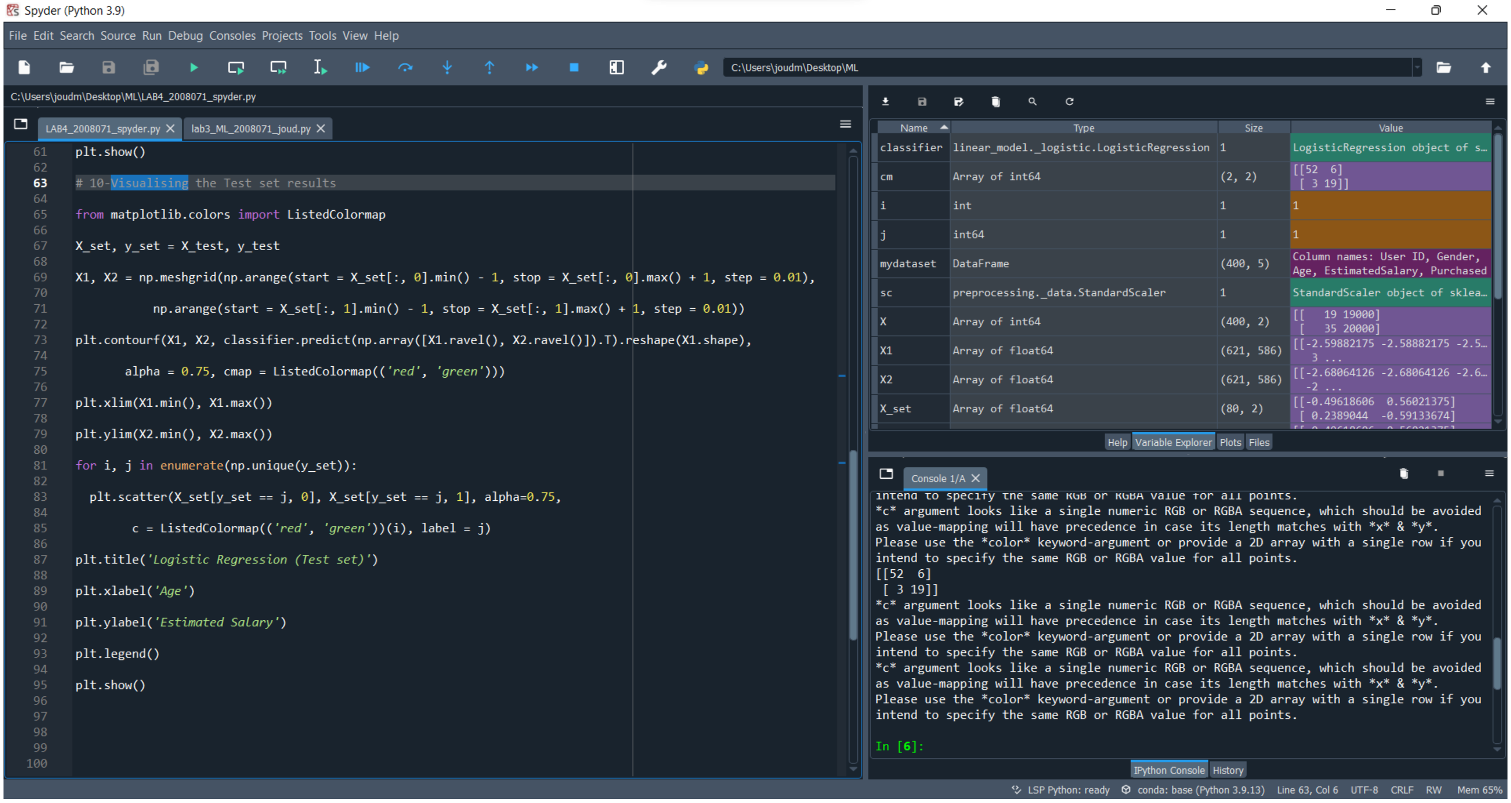
```python
X_test = sc.fit_transform(X_test)


# 6-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)


# 7-Predicting the Test set results
y_pred = classifier.predict(X_test)


# 8-Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)


# 9-Visualising the Training set results
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
           np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(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], alpha=0.75,
        c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Logistic Regression (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

| Name | Type | Size | Value |
|---|---|---|---|
| classifier | linear_model._logistic.LogisticRegression | 1 | LogisticRegression object of sklea… |
| cm | Array of int64 | (2, 2) | [[52  6]<br> [ 3 19]] |
| i | int | 1 | 1 |
| j | int64 | 1 | 1 |
| mydataset | DataFrame | (400, 5) | Column names: User ID, Gender, Age, EstimatedSalary, Purchased |
| sc | preprocessing._data.StandardScaler | 1 | StandardScaler object of sklearn.preprocessing._data module |
| X | Array of int64 | (400, 2) | [[   19 19000]<br> [   35 20000] |
| X1 | Array of float64 | (592, 609) | [[-2.96547978 -2.95547978 -2.94547…<br>3 ... |
| X2 | Array of float64 | (592, 609) | [[-2.59138156 -2.59138156 -2.59138…<br>-2 ... |
| X_set | Array of float64 | (320, 2) | [[ 1.92295008  2.14601566]<br> [ 2.02016082  0.3787193 ] |

```
*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.
[[52  6]
 [ 3 19]]

                              Warning
 Figures now render in the Plots pane by default. To make them also appear
 inline in the Console, uncheck "Mute Inline Plotting" under the Plots pane
 options menu.

In [5]:
```



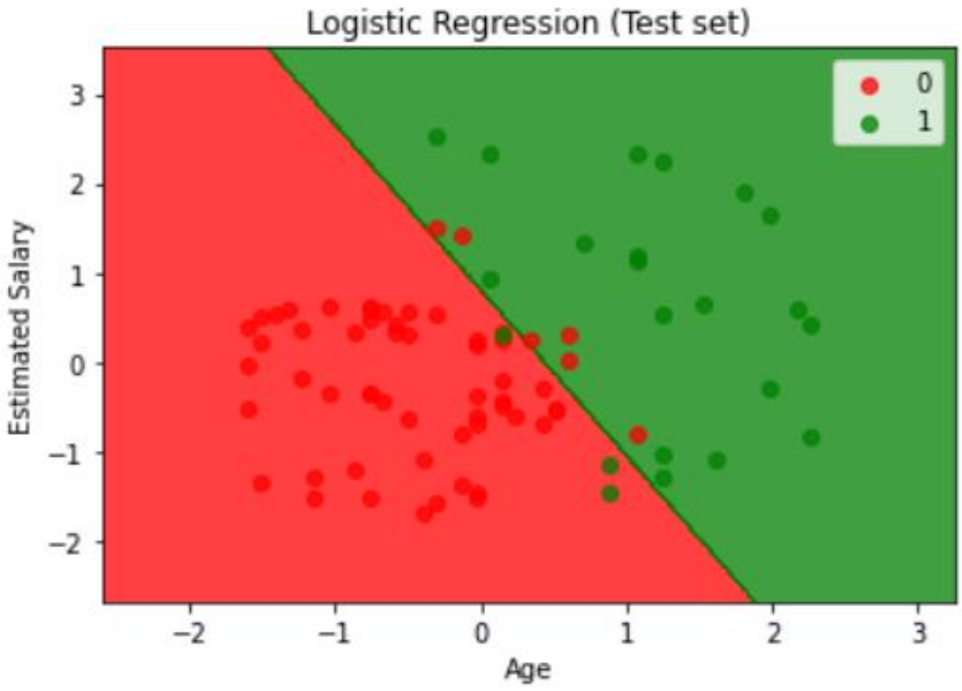
# Visualising the Test set results



```
plt.show()

# 10-Visualising the Test set results

from matplotlib.colors import ListedColormap

X_set, y_set = X_test, y_test

X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),

           np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))

plt.contourf(X1, X2, classifier.predict(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], alpha=0.75,

        c = ListedColormap(('red', 'green'))(i), label = j)

plt.title('Logistic Regression (Test set)')

plt.xlabel('Age')

plt.ylabel('Estimated Salary')

plt.legend()

plt.show()
```

| Name | Type | Size | Value |
|---|---|---|---|
| classifier | linear_model._logistic.LogisticRegression | 1 | LogisticRegression object of s… |
| cm | Array of int64 | (2, 2) | [[52  6]<br> [ 3 19]] |
| i | int | 1 | 1 |
| j | int64 | 1 | 1 |
| mydataset | DataFrame | (400, 5) | Column names: User ID, Gender, Age, EstimatedSalary, Purchased |
| sc | preprocessing._data.StandardScaler | 1 | StandardScaler object of sklea… |
| X | Array of int64 | (400, 2) | [[   19 19000]<br> [   35 20000] |
| X1 | Array of float64 | (621, 586) | [[-2.59882175 -2.58882175 -2.5…<br>  3 ... |
| X2 | Array of float64 | (621, 586) | [[-2.68064126 -2.68064126 -2.6…<br>  -2 ... |
| X_set | Array of float64 | (80, 2) | [[-0.49618606  0.56021375]<br> [ 0.2389044  -0.59133674] |

```
intend to specify the same RGB or RGBA value for all points.
*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.
[[52  6]
 [ 3 19]]
*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.
*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.

In [6]:
```


Logistic Regression (Test set)
Estimated Salary
Age
● 0
● 1