In [37]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as mtp
```

In [ ]:
```
import pandas as pd
data= pd.read_csv("C:\\Users\\sriha\\OneDrive\\Desktop\\pb excel\\salary.csv")
print(data)
```

In [13]:
```
x= data.iloc[:,[1,2]]
y= data.iloc[:,3]
print(x,y)
```

```
   Experience  Salary
0           5   50000
1           3   24000
2           4   45000
3           1   19000
4           7   80000
5           4   50000
6           4   43000
7           8   97000 0    0
1    1
2    1
3    0
4    1
5    1
6    1
7    1
Name: binary, dtype: int64
```

In [14]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y, test_size= 0.25 , random_sta
```

In [ ]:
```
#feature Scaling
from sklearn.preprocessing import StandardScaler
st_x= StandardScaler()
x_train= st_x.fit_transform(x_train)
x_test= st_x.transform(x_test)
```

In [16]:
```
#Fitting Logistic Regression to the training set
from sklearn.linear_model import LogisticRegression
classifier= LogisticRegression(random_state=0)
classifier.fit(x_train, y_train)
```

Out[16]:
```
LogisticRegression(random_state=0)
```

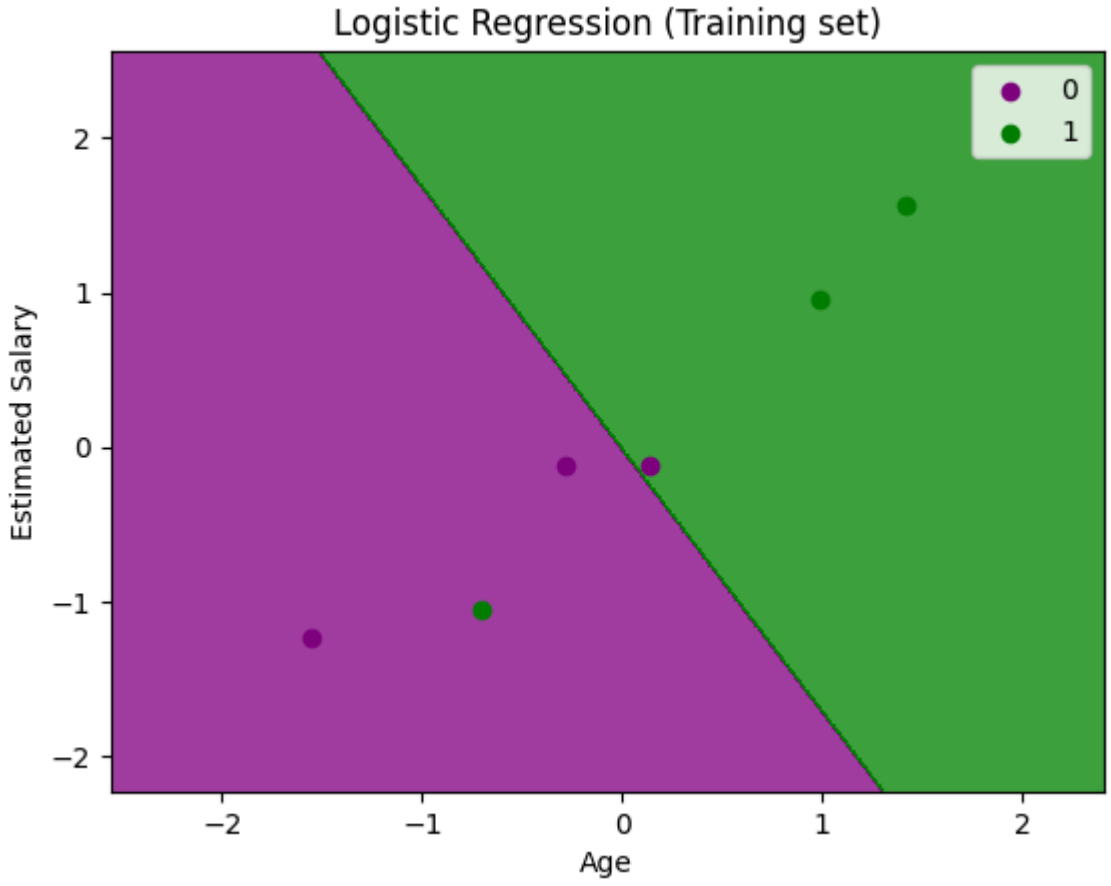
In [30]:
```
#Predicting the test set result
y_pred= classifier.predict(x_test)
```

In [36]:
```
#Creating the Confusion matrix
from sklearn.metrics import confusion_matrix
cm= confusion_matrix(y_test,y_pred)
```

In [44]:
```
#Visualizing the training set result
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]
```

```
np.arange(start = x_set[:, 1].min() - 1, stop = x_set[:, 1].max() + 1, step = 0.
mtp.contourf(x1, x2, classifier.predict(np.array([x1.ravel(), x2.ravel()]).T).re
alpha = 0.75, cmap = ListedColormap(('purple','green' )))
mtp.xlim(x1.min(), x1.max())
mtp.ylim(x2.min(), x2.max())
for i, j in enumerate(np.unique(y_set)):
    mtp.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1],
        c = ListedColormap(('purple', 'green'))(i), label = j)
mtp.title('Logistic Regression (Training set)')
mtp.xlabel('Age')
mtp.ylabel('Estimated Salary')
mtp.legend()
mtp.show()
```

```
*c* argument looks like a single numeric RGB or RGBA sequence, which should be av
oided 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 av
oided 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 [ ]: