

```

#Data Pre-procesing Step
# importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd

#importing datasets
data_set= pd.read_csv('/content/drive/MyDrive/Machine Learning/ML Experiments/Experient No 3/User_Data.csv')

#Extracting Independent and dependent Variable
x= data_set.iloc[:, [2,3]].values
y= data_set.iloc[:, 4].values

# Replacing commas in numerical columns and converting to numeric type
x[:, 1] = [float(str(i).replace(',',' ')) for i in x[:, 1]] # Assuming the second column (index 1) contains the numerical values with commas

# Splitting the dataset into training 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= 0.25, random_state=0)

#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)

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



↔

LogisticRegression
▼



LogisticRegression(random_state=0)




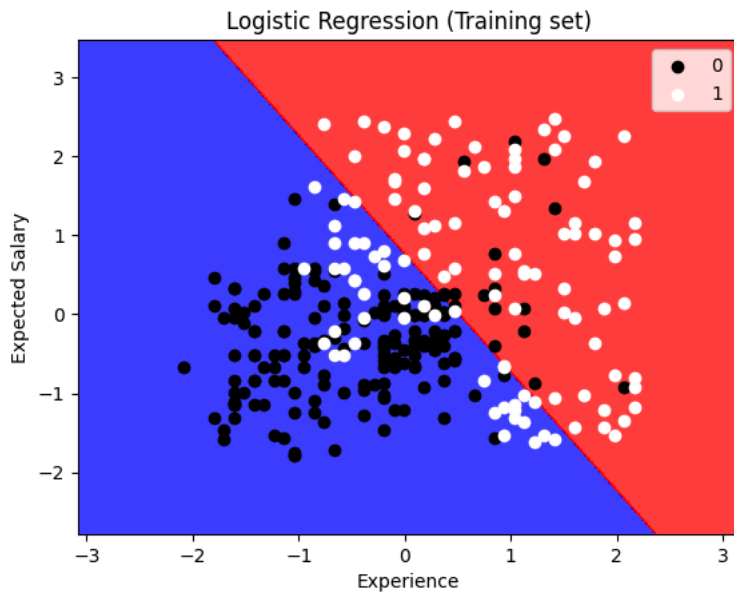
#Predicting the test set result
y_pred= classifier.predict(x_test)

#Creating the Confusion matrix
from sklearn.metrics import confusion_matrix
cm= confusion_matrix(y_test, y_pred) # Pass the true and predicted labels


#Visualizing the training set result
from matplotlib.colors import ListedColormap
x_set, y_set = x_train, y_train
x1, x2 = nm.meshgrid(nm.arange(start = x_set[:, 0].min() - 1, stop = x_set[:, 0].max() + 1, step =0.01),
nm.arange(start = x_set[:, 1].min() - 1, stop = x_set[:, 1].max() + 1, step = 0.01))
mtp.contourf(x1, x2, classifier.predict(nm.array([x1.ravel(), x2.ravel()]).T).reshape(x1.shape),
alpha = 0.75, cmap = ListedColormap(('blue','red' )))
mtp.xlim(x1.min(), x1.max())
mtp.ylim(x2.min(), x2.max())
for i, j in enumerate(nm.unique(y_set)):
    mtp.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1],
        c = ListedColormap(('black', 'white'))(i), label = j)
mtp.title('Logistic Regression (Training set)')
mtp.xlabel('Experience')
mtp.ylabel('Expected Salary')
mtp.legend()
mtp.show()

```

 <ipython-input-43-e9cd442459d2>:11: UserWarning: *c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided
 mtp.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1],



```
#Visulaizing the test set result
from matplotlib.colors import ListedColormap
x_set, y_set = x_test, y_test
x1, x2 = nm.meshgrid(nm.arange(start = x_set[:, 0].min() - 1, stop = x_set[:, 0].max() + 1, step = 0.01),
nm.arange(start = x_set[:, 1].min() - 1, stop = x_set[:, 1].max() + 1, step = 0.01))
mtp.contourf(x1, x2, classifier.predict(nm.array([x1.ravel(), x2.ravel()]).T).reshape(x1.shape),
alpha = 0.75, cmap = ListedColormap(('blue', 'red' )))
mtp.xlim(x1.min(), x1.max())
mtp.ylim(x2.min(), x2.max())
for i, j in enumerate(nm.unique(y_set)):
    mtp.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1],
        c = ListedColormap(('black', 'white'))(i), label = j)
mtp.title('Logistic Regression (Test set)')
mtp.xlabel('Experience')
mtp.ylabel('Expected Salary')
mtp.legend()
mtp.show()
```

 <ipython-input-44-4eef9c92a5af>:11: UserWarning: *c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided
 mtp.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1],

