```
# Logistic Regression

# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv(r"C:\Users\DELL\Downloads\final1.csv")
#this datasset contian information of user and socianl network, those features are - userid,gender,age,salary,purchased
#social network has several business client which can put their into social networks and one of the client is car company , this company has newly lunched XUV in rediculous price or high price
#we will see which of the user in this social network are going to buy brand new xuv car
#Last column tell us user purchased the car yes-1 // no-0 & we are going to build the model that is goint to predict if the user is going to buy xuv or not based on 2 variable based on age & estimated salery
#so our matrix of feature is only these 2 column & we gonna find some corelation b/w age and estimated salary of user and his decission to purchase the car [yes or no]
#so i need 2 index and rest of index i will remove for this i have to use slicing operator
#1 means - the user going to buy the car & 0 means - user is not going to buy the car

X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, -1].values

# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
#for this observation let me selcted as 100 observaion for test set
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)


#we are going to predict which users are going to predit xuv,


# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
#we mentioned feature scaling only to independent variable not dependent variable at all


#datapreprocessing done guys upto this part


#**************************************************************************
****************


#Next step is we are going to build the logistic model and appy this model into our dataset
#This is linear model library thats why we called from sklear.linear_model


# Training the Logistic Regression model on the Training set


from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
classifier.fit(X_train, y_train)
#we have to fit the logistic regression model to our training set


# Predicting the Test set results
```

```
y_pred = classifier.predict(X_test)

#now you compare X_test with y_pred, x-test we ha,ve age and salary ,

#if u look at the first observation this user is not be able to buy the car but if you look at observation 7 then that user is going to buy the car

#in this case logistic regression model classify the which users are going to buy the car or not


#we build our logistic model and fit it to the training set & we predict our test set result


from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, y_pred)

cm


from sklearn.metrics import accuracy_score

ac = accuracy_score(y_test, y_pred)

ac


# This is to get the Classification Report

from sklearn.metrics import classification_report

cr = classification_report(y_test, y_pred)

cr


bias = classifier.score(X_train,y_train)

bias


variance = classifier.score(X_test, y_test)

variance


#-----------------FUTURE PREDICTION ------------
```

```
dataset1 = pd.read_csv(r"C:\Users\Admin\Desktop\MyFile\0. DATASCIENCE PROJECT\15. Logistic regression with future prediction\Future prediction1.csv")

d2 = dataset1.copy()

dataset1 = dataset1.iloc[:, [2, 3]].values

from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
M = sc.fit_transform(dataset1)

y_pred1 = pd.DataFrame()

d2 ['y_pred1'] = classifier.predict(M)

d2.to_csv('final1.csv')

'''
# To get the path
import os
os.getcwd()
'''
#--------
'''
dataset2 = pd.read_csv(r"C:\Users\kdata\Desktop\KODI WORK\1. NARESH\1. MORNING BATCH\N_Batch -- 10.00AM\3. May\26th,27th\Future prediction1.csv")

dataset2 = dataset1.iloc[:, [2, 3]].values
```

```
dataset2.to_csv(classfier)


df_final.to_csv('y_pred1.to_csv',index=False)


#df_final.to_csv('data_final.csv',index=False)


data_test.to_csv('final.csv')
'''


#*************************************************************************
******************************************
```