Chapter 3: Demo Logistic Regression - Buy car?

- Xây dựng model dự đoán một khách hàng có mua xe hay không dựa trên thông tin về 'Age' và 'EstimatedSalary_K' (mức lương ước tính – đơn vị tính 1000$)

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [ ]:
```
# Dataset: https://www.kaggle.com/rakeshrau/social-network-ads
```

In [2]:
```
data = pd.read_csv("Social_Network_Ads.csv",
                   usecols=['Age', 'EstimatedSalary_K', 'Purchased'])
```

In [3]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 3 columns):
Age                 400 non-null int64
EstimatedSalary_K   400 non-null int64
Purchased           400 non-null int64
dtypes: int64(3)
memory usage: 9.5 KB
```

In [4]:
```
data.head()
```

Out[4]:

	Age	EstimatedSalary_K	Purchased
0	19	19	0
1	35	20	0
2	26	43	0
3	27	57	0
4	19	76	0

In [5]:
```
data.describe()
```

Out[5]:

	Age	EstimatedSalary_K	Purchased
count	400.000000	400.00000	400.000000
mean	37.655000	69.74250	0.357500
std	10.482877	34.09696	0.479864
min	18.000000	15.00000	0.000000
25%	29.750000	43.00000	0.000000
50%	37.000000	70.00000	0.000000
75%	46.000000	88.00000	1.000000
max	60.000000	150.00000	1.000000

In [6]:
```
X = data[['Age', 'EstimatedSalary_K']]
X.head()
```

Out[6]:

	Age	EstimatedSalary_K
0	19	19
1	35	20
2	26	43
3	27	57
4	19	76

In [7]:
```
Y = data['Purchased']
Y.head()
```

Out[7]:
```
0    0
1    0
2    0
3    0
4    0
Name: Purchased, dtype: int64
```

In [8]:
```
from sklearn.model_selection import train_test_split
```

In [9]:
```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.3)
```

In [10]:
```
from sklearn.linear_model import LogisticRegression
```

In [11]:
```
clf = LogisticRegression()
```

In [12]:
```
clf.fit(X_train, Y_train)
```

```
c:\program files\python36\lib\site-packages\sklearn\linear_model\logistic.py:43
2: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify a
solver to silence this warning.
  FutureWarning)
```

Out[12]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='warn', n_jobs=None, penalty='l2',
                   random_state=None, solver='warn', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [13]:
```
# Tính toán xác suất của lớp cho tập dữ liệu thử nghiệm
# bằng cách sử dụng hàm 'predict_proba'.
# clf.predict_proba(X_test)
```

In [14]:
```
print('Train score: ', clf.score(X_train,Y_train))
```

```
Train score:  0.8428571428571429
```

In [15]:
```
print('Test score: ', clf.score(X_test,Y_test))
```

```
Test score:  0.8333333333333334
```

In [16]:
```
Yhat_train = clf.predict(X_train)
```

In [17]:
```
Yhat_test = clf.predict(X_test)
Yhat_test
```

Out[17]:
```
array([0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1,
       1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0,
       0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1,
       0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=int64)
```

In [18]:
```
from sklearn.metrics import accuracy_score
```

In [19]:
```
print("Test Accuracy is ", accuracy_score(Y_test,Yhat_test)*100,"%")
```

```
Test Accuracy is  83.33333333333334 %
```

In [20]:
```
from sklearn.metrics import confusion_matrix
```

In [21]:
```
cm = confusion_matrix(Y_test, Yhat_test)
```

In [22]:
```
cm
```

Out[22]:
```
array([[75,  5],
       [15, 25]], dtype=int64)
```

In [23]:
```
clf.intercept_
```

Out[23]:
```
array([-5.81655155])
```

In [24]:
```
clf.coef_
#tuong ung voi intercept,
```

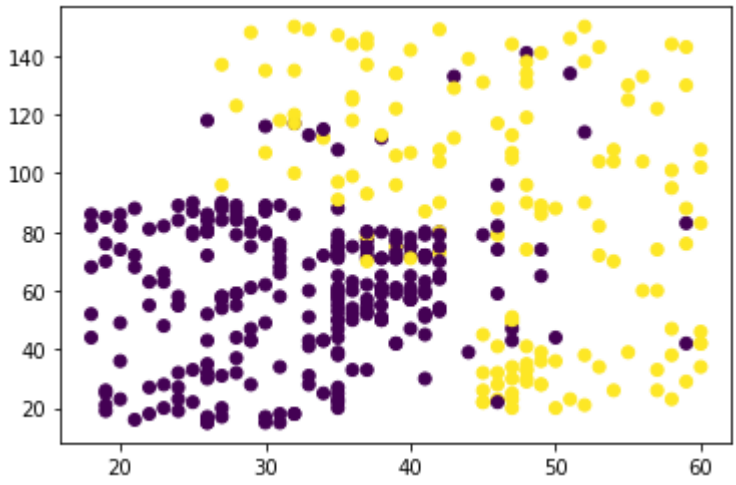
Out[24]:
```
array([[0.10880121, 0.01597048]])
```

Trực quan hóa 1

In [25]:
```
import seaborn as sns
```

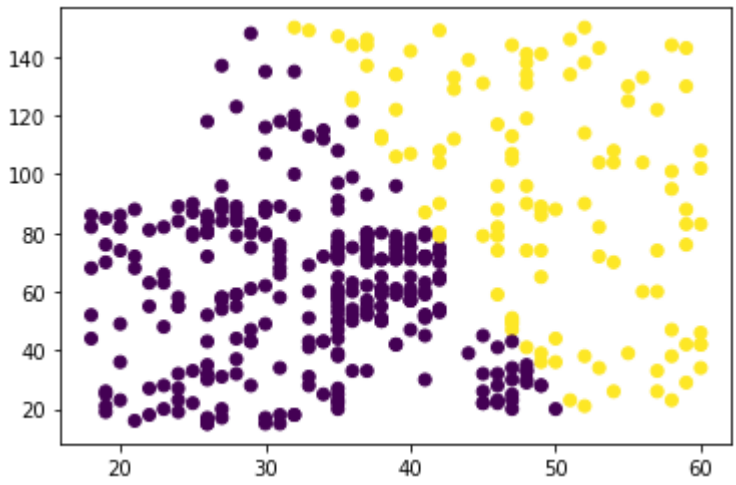
In [26]:
```
plt.scatter(X.Age , X.EstimatedSalary_K, c= Y)
```

Out[26]:
```
<matplotlib.collections.PathCollection at 0x2805fbdd828>
```

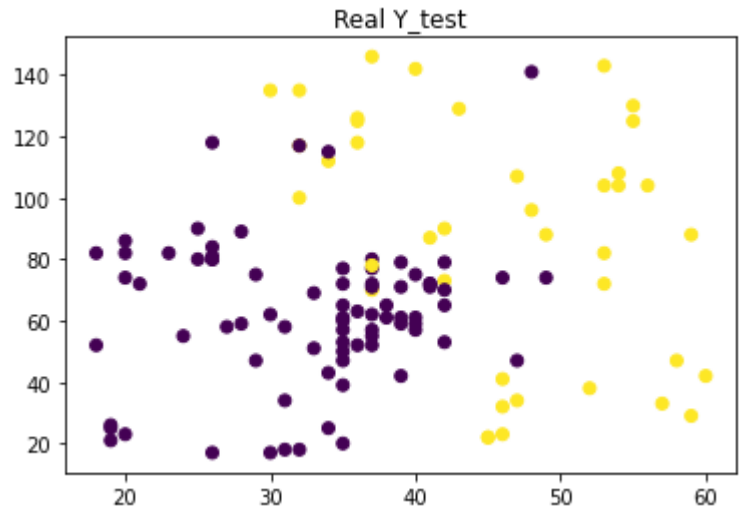


In [27]:
```
plt.scatter(X.Age , X.EstimatedSalary_K, c= clf.predict(X))
```

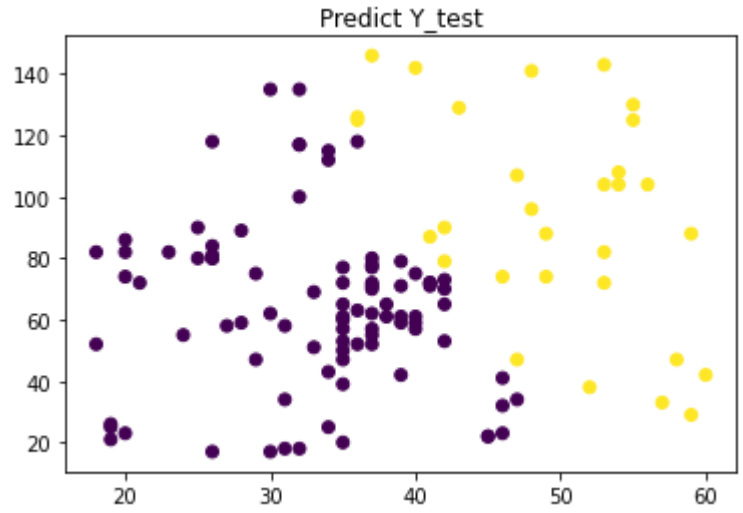
Out[27]:
```
<matplotlib.collections.PathCollection at 0x2805fcd0400>
```



In [33]:
```
plt.scatter(X_test.Age , X_test.EstimatedSalary_K,
            c=Y_test)
plt.title('Real Y_test')
plt.show()
```



In [34]:
```
plt.scatter(X_test.Age , X_test.EstimatedSalary_K,
            c=Yhat_test)
plt.title('Predict Y_test')
plt.show()
```



Trực quan hóa 2

In [43]:
```
from matplotlib.colors import ListedColormap
```

In [39]:
```
X_set, Y_set = X_test, Y_test
```

In [40]:
```
X_set.head()
```

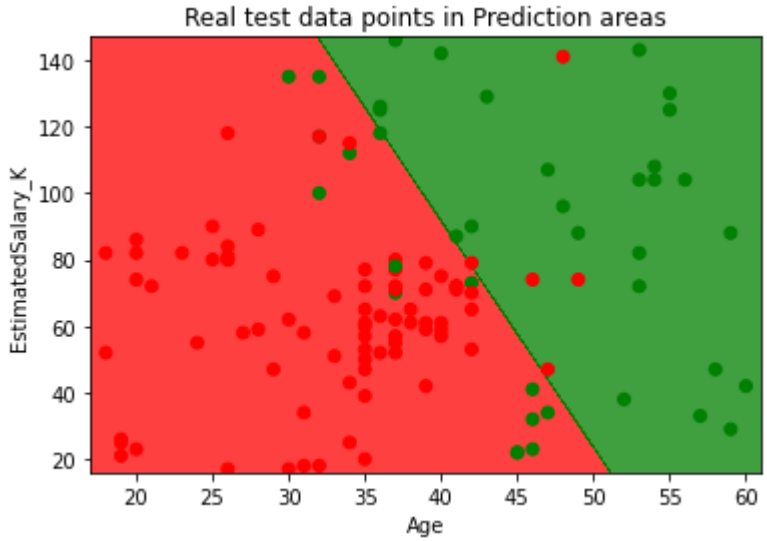
Out[40]:

	Age	EstimatedSalary_K
159	32	135
76	18	52
191	19	26
195	34	43
329	47	107

In [46]:
```
np.unique(Y_set)
```

Out[46]:
```
array([0, 1], dtype=int64)
```

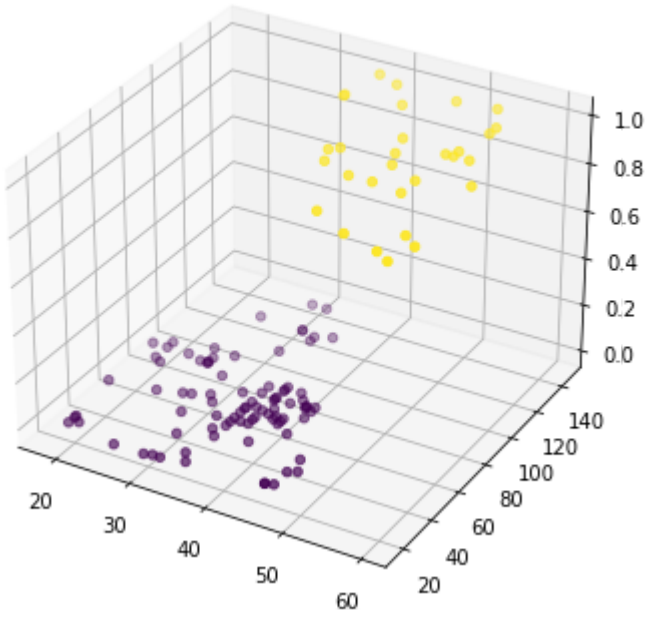
In [55]:
```
# so sánh giữa real test (điểm dữ liệu) và prediction test (phần xanh/đỏ)
from matplotlib.colors import ListedColormap
X_set, Y_set = X_test, Y_test
X1, X2 = np.meshgrid(np.arange(start = X_set.Age.min()-1,stop=X_set.Age.max()+1,s
                     np.arange(start = X_set.EstimatedSalary_K.min()-1,stop=X_set
                               step=0.01))
plt.contourf(X1,X2, clf.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1
             alpha=0.75, cmap=ListedColormap(('red','green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
plt.scatter(X_set.Age, X_set.EstimatedSalary_K, c=Y_set, cmap=ListedColormap(('re
plt.xlabel('Age')
plt.ylabel('EstimatedSalary_K')
plt.title('Real test data points in Prediction areas')
plt.show()
```



Trực quan hóa 3

In [30]:
```
from mpl_toolkits.mplot3d import Axes3D
```

In [31]:
```
fig = plt.figure(figsize=(6,6))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(X_test.Age, X_test.EstimatedSalary_K, Yhat_test,
           c=Yhat_test)
plt.show()
```



In [32]:
```
X_now = [[40,120]]
Y_now = clf.predict(X_now)
Y_now
```

Out[32]:
```
array([1], dtype=int64)
```