```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd


dataset = pd.read_csv('Social_Network_Ads.csv')
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values


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)


print(X_train)
```

```
[    21  72000]
[    38  71000]
[    39 106000]
[    37  57000]
[    26  72000]
[    35  23000]
[    54 108000]
[    30  17000]
[    39 134000]
[    29  43000]
[    33  43000]
[    35  38000]
[    41  45000]
[    41  72000]
[    39 134000]
[    27 137000]
[    21  16000]
[    26  32000]
[    31  66000]
[    39  73000]
[    41  79000]
[    47  50000]
[    41  30000]
[    37  93000]
[    60  46000]
[    25  22000]
[    28  37000]
[    38  55000]
[    36  54000]
[    20  36000]
[    56 104000]
[    40  57000]
[    42 108000]
[    20  23000]
[    40  65000]
[    47  20000]
[    18  86000]
[    35  79000]
[    57  33000]
```

```
[    34  72000]
[    49  39000]
[    27  31000]
[    19  70000]
[    39  79000]
[    26  81000]
[    25  80000]
[    28  85000]
[    55  39000]
[    50  88000]
[    49  88000]
[    52 150000]
[    35  65000]
[    42  54000]
[    34  43000]
[    37  52000]
[    48  30000]
[    29  43000]
[    36  52000]
```

```
print(X_test)
```

```
[    27  84000]
[    35  20000]
[    43 112000]
[    27  58000]
[    37  80000]
[    52  90000]
[    26  30000]
[    49  86000]
[    57 122000]
[    34  25000]
[    35  57000]
[    34 115000]
[    59  88000]
[    45  32000]
[    29  83000]
[    26  80000]
[    49  28000]
[    23  20000]
[    32  18000]
[    60  42000]
[    19  76000]
[    36  99000]
[    19  26000]
[    60  83000]
[    24  89000]
[    27  58000]
[    40  47000]
[    42  70000]
[    32 150000]
[    35  77000]
[    22  63000]
[    45  22000]
[    27  89000]
```

```
[    18  82000]
[    42  79000]
[    40  60000]
[    53  34000]
[    47 107000]
[    58 144000]
[    59  83000]
[    24  55000]
[    26  35000]
[    58  38000]
[    42  80000]
[    40  75000]
[    59 130000]
[    46  41000]
[    41  60000]
[    42  64000]
[    37 146000]
[    23  48000]
[    25  33000]
[    24  84000]
[    27  96000]
[    23  63000]
[    48  33000]
[    48  90000]
[    42 104000]]
```

```
print(y_test)
```

```
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 1 0 0 0 0 0 1 1 0 0 0 0
 0 0 1 0 0 0 0 1 0 0 1 0 1 1 0 0 0 1 1 0 0 1 0 0 1 0 1 0 1 0 0 0 0 1 0 0 1
 0 0 0 0 1 1 1 0 0 0 1 1 0 1 1 0 0 1 0 0 0 1 0 1 1 1]
```

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
print(X_train)
```

```
[-1.6960924   0.07006676]
[-0.01254409  0.04107362]
[ 0.08648817  1.05583366]
[-0.11157634 -0.3648304 ]
[-1.20093113  0.07006676]
[-0.30964085 -1.3505973 ]
[ 1.57197197  1.11381995]
[-0.80480212 -1.52455616]
[ 0.08648817  1.8676417 ]
[-0.90383437 -0.77073441]
[-0.50770535 -0.77073441]
[-0.30964085 -0.91570013]
[ 0.28455268 -0.71274813]
[ 0.28455268  0.07006676]
[ 0.08648817  1.8676417 ]
[-1.10189888  1.95462113]
```

```
[-1.6960924  -1.5535493 ]
[-1.20093113 -1.089659  ]
[-0.70576986 -0.1038921 ]
[ 0.08648817  0.09905991]
[ 0.28455268  0.27301877]
[ 0.8787462  -0.5677824 ]
[ 0.28455268 -1.14764529]
[-0.11157634  0.67892279]
[ 2.1661655  -0.68375498]
[-1.29996338 -1.37959044]
[-1.00286662 -0.94469328]
[-0.01254409 -0.42281668]
[-0.21060859 -0.45180983]
[-1.79512465 -0.97368642]
[ 1.77003648  0.99784738]
[ 0.18552042 -0.3648304 ]
[ 0.38358493  1.11381995]
[-1.79512465 -1.3505973 ]
[ 0.18552042 -0.13288524]
[ 0.8787462  -1.43757673]
[-1.99318916  0.47597078]
[-0.30964085  0.27301877]
[ 1.86906873 -1.06066585]
[-0.4086731   0.07006676]
[ 1.07681071 -0.88670699]
[-1.10189888 -1.11865214]
[-1.89415691  0.01208048]
[ 0.08648817  0.27301877]
[-1.20093113  0.33100506]
[-1.29996338  0.30201192]
[-1.00286662  0.44697764]
[ 1.67100423 -0.88670699]
[ 1.17584296  0.53395707]
[ 1.07681071  0.53395707]
[ 1.37390747  2.331532  ]
[-0.30964085 -0.13288524]
[ 0.38358493 -0.45180983]
[-0.4086731  -0.77073441]
[-0.11157634 -0.50979612]
[ 0.97777845 -1.14764529]
[-0.90383437 -0.77073441]
[-0.21060859 -0.50979612]
```

```
print(X_test)
```

```
[[-0.80480212  0.50496393]
 [-0.01254409 -0.5677824 ]
 [-0.30964085  0.1570462 ]
 [-0.80480212  0.27301877]
 [-0.30964085 -0.5677824 ]
 [-1.10189888 -1.43757673]
 [-0.70576986 -1.58254245]
 [-0.21060859  2.15757314]
 [-1.99318916 -0.04590581]
```

```
       [ 0.8787462  -0.77073441]
       [-0.80480212 -0.59677555]
       [-1.00286662 -0.42281668]
       [-0.11157634 -0.42281668]
       [ 0.08648817  0.21503249]
       [-1.79512465  0.47597078]
       [-0.60673761  1.37475825]
       [-0.11157634  0.21503249]
       [-1.89415691  0.44697764]
       [ 1.67100423  1.75166912]
       [-0.30964085 -1.37959044]
       [-0.30964085 -0.65476184]
       [ 0.8787462   2.15757314]
       [ 0.28455268 -0.53878926]
       [ 0.8787462   1.02684052]
       [-1.49802789 -1.20563157]
       [ 1.07681071  2.07059371]
       [-1.00286662  0.50496393]
       [-0.90383437  0.30201192]
       [-0.11157634 -0.21986468]
       [-0.60673761  0.47597078]
       [-1.6960924   0.53395707]
       [-0.11157634  0.27301877]
       [ 1.86906873 -0.27785096]
       [-0.11157634 -0.48080297]
       [-1.39899564 -0.33583725]
       [-1.99318916 -0.50979612]
       [-1.59706014  0.33100506]
       [-0.4086731  -0.77073441]
       [-0.70576986 -1.03167271]
       [ 1.07681071 -0.97368642]
       [-1.10189888  0.53395707]
       [ 0.28455268 -0.50979612]
       [-1.10189888  0.41798449]
       [-0.30964085 -1.43757673]
       [ 0.48261718  1.22979253]
       [-1.10189888 -0.33583725]
       [-0.11157634  0.30201192]
       [ 1.37390747  0.59194336]
       [-1.20093113 -1.14764529]
       [ 1.07681071  0.47597078]
       [ 1.86906873  1.51972397]
       [-0.4086731  -1.29261101]
       [-0.30964085 -0.3648304 ]
       [-0.4086731   1.31677196]
       [ 2.06713324  0.53395707]
       [ 0.68068169 -1.089659  ]
       [-0.90383437  0.38899135]
```

```
from sklearn.svm import SVC
classifier = SVC(kernel = 'linear', random_state = 0)
classifier.fit(X_train, y_train)
```

```
SVC(kernel='linear', random_state=0)
```

```
print(classifier.predict(sc.transform([[30,87000]])))
```

```
[0]
```

```
y_pred = classifier.predict(X_test)
print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))
```

```
[0 0]
[0 0]
[1 1]
[0 0]
[0 0]

[1 1]
[0 0]
[1 1]
[1 1]
[0 0]
[0 0]
[0 0]
[1 1]
[0 1]
[0 0]
[0 0]
[0 1]
[0 0]
[0 0]
[1 1]
[0 0]
[0 1]
[0 0]
[1 1]
[0 0]
[0 0]
[0 0]
[0 0]
[1 1]
[0 0]
[0 0]
[0 1]
[0 0]
[0 0]
[1 0]
[0 0]
[1 1]
[1 1]
[1 1]
[1 0]
[0 0]
[0 0]
[1 1]
[1 1]
[0 0]
[1 1]
[0 1]
[0 0]
```

```
[0 0]
[0 0]
[1 1]
[0 0]
[0 0]
[0 0]
[0 1]
[0 0]
[0 1]
[1 1]
[1 1]]
```

```
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
```

```
[[66  2]
 [ 8 24]]
0.9
```

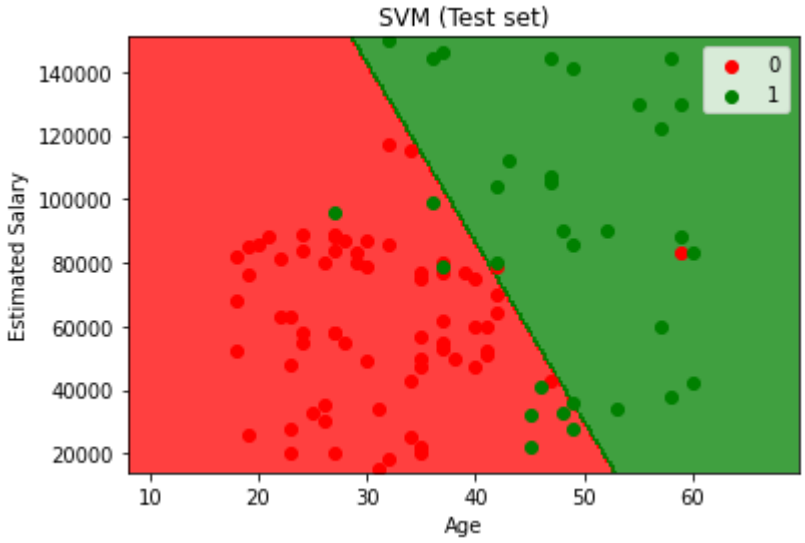
```
from matplotlib.colors import ListedColormap
X_set, y_set = sc.inverse_transform(X_train), y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 10, stop = X_set[:, 0].max() + 10,
                     np.arange(start = X_set[:, 1].min() - 1000, stop = X_set[:, 1].max() + 1
plt.contourf(X1, X2, classifier.predict(sc.transform(np.array([X1.ravel(), X2.ravel()]).T)).r
             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], c = ListedColormap(('red', 'greer
plt.title('SVM (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

```
*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided
*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided
```

SVM (Training set)

```
from matplotlib.colors import ListedColormap
X_set, y_set = sc.inverse_transform(X_test), y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 10, stop = X_set[:, 0].max() + 10,
                     np.arange(start = X_set[:, 1].min() - 1000, stop = X_set[:, 1].max() + 1
plt.contourf(X1, X2, classifier.predict(sc.transform(np.array([X1.ravel(), X2.ravel()]).T)).r
             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], c = ListedColormap(('red', 'greer
plt.title('SVM (Test set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

```
*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided
*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided
```



✓ 6m 13s completed at 11:18 AM