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

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
dataset=pd.read_csv("/content/drive/MyDrive/Personal/Studies/MSC Data Science Material/SEM2/ML/Practical/data_set/Social_Network_Ads.csv
```

```
dataset
```

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0
...	...	...	...	...	...
395	15691863	Female	46	41000	1
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

400 rows × 5 columns

```
#Splitting dataset into X and Y
x=dataset.iloc[:,[2,3]].values
y=dataset.iloc[:,4].values
```

```
#Splitting X and Y dataset into Train and Test data
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)
```

```
x_train
```

```
[    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],
[    27,  54000],
[    26. 118000]])
```

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]])
```

y_train

```
array([0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1,
       0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0,
       0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0,
       0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0,
       1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1,
       0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0,
       1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0,
       0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0,
       1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0,
       0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0,
       1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1,
       1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0,
       0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0])
```

```
y_test
```

```
array([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])
```

```
#Perform Feature scaling
#in the dataset all values are not in the same range hence we use feature scaling to overcome this problem
#feature scaling helps us normalize data within the range

#fit_transform will fit all the data in the variable

from sklearn.preprocessing import StandardScaler

sc=StandardScaler()
X_Train=sc.fit_transform(x_train)
X_Test=sc.fit_transform(x_test)
```

```
X_Train
```

```
[ 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],
[-1.10189888, -0.45180983],
[-1.20093113,  1.40375139]])
```

X_Test

```
[-0.81070599,  0.42134337],
[-0.10879604, -1.53554892],
[ 0.59311391,  1.27748375],
[-0.81070599, -0.37364412],
[ 0.06668145,  0.2990376 ],
[ 1.3827626 ,  0.60480202],
[-0.89844474, -1.2297845 ],
[ 1.11954637,  0.48249625],
[ 1.82145632,  1.58324817],
[-0.19653479, -1.38266671],
[-0.10879604, -0.40422056],
[-0.19653479,  1.36921307],
[ 1.99693381,  0.54364914],
[ 0.7685914 , -1.16863161],
[-0.63522851,  0.39076693],
[-0.89844474,  0.2990376 ],
[ 1.11954637, -1.29093738],
[-1.16166097, -1.53554892],
[-0.37201227, -1.5967018 ],
[ 2.08467255, -0.86286719],
[-1.51261594,  0.17673183],
[-0.0210573 ,  0.87999   ],
[-1.51261594, -1.35209027],
[ 2.08467255,  0.39076693],
[-1.07392223,  0.57422558],
[-0.81070599, -0.37364412],
[ 0.32989768, -0.70998498],
[ 0.50537516, -0.00672682],
[-0.37201227,  2.43938854],
[-0.10879604,  0.20730828],
[-1.24939971, -0.22076191],
[ 0.7685914 , -1.47439603],
[-0.81070599,  0.57422558],
[-1.60035469,  0.36019049],
[ 0.50537516,  0.26846116],
[ 0.32989768, -0.31249124],
[ 1.47050135, -1.10747873],
[ 0.94406888,  1.12460154],
[ 1.90919507,  2.25592989],
[ 1.99693381,  0.39076693],
[-1.07392223, -0.46537345],
[-0.89844474, -1.07690229],
[ 1.90919507, -0.98517296],
[ 0.50537516,  0.2990376 ],
[ 0.32989768,  0.14615539],
[ 1.99693381,  1.8278597 ],
[ 0.85633014, -0.89344364],
[ 0.41763642, -0.31249124],
[ 0.50537516, -0.19018547],
[ 0.06668145,  2.31708278],
[-1.16166097, -0.67940854],
[-0.98618348, -1.13805517],
[-1.07392223,  0.42134337],
[-0.81070599,  0.78826068],
[-1.16166097, -0.22076191],
[ 1.03180763, -1.13805517],
[ 1.03180763,  0.60480202],
[ 0.50537516,  1.03287221]])
```

```
#We are done with data pre-processing


#Fit SVM to training set

from sklearn.svm import SVC
classifier=SVC(kernel="rbf", random_state=0)
classifier.fit(X_Train, y_train)
```

▾ SVC

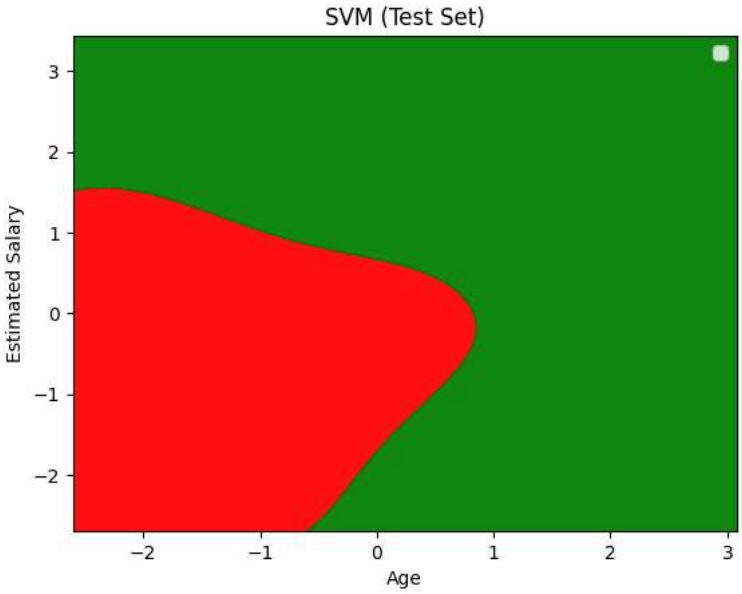
SVC(random_state=0)

```
#Predict the test set result
y_pred= classifier.predict(X_Test)


#Make the confusion matrix

from sklearn.metrics import confusion_matrix, accuracy_score
cm=confusion_matrix(y_test, y_pred)
print(cm)
print("\n Accuracy: ")
accuracy_score(y_test,y_pred)
```

```
[[64  4]
 [ 3 29]]

 Accuracy:
0.93
```

```
#Visualise the test set results

from matplotlib.colors import ListedColormap

X_set, y_set = X_Test, y_test
X1, X2 = np.meshgrid(np.arange(start=X_set[:,0].min() - 1, stop=X_set[:,0].max() + 1, step=0.01),
                     np.arange(start=X_set[:,1].min() - 1, stop=X_set[:,1].max() + 1, step=0.01))
plt.contourf(X1,X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             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.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha=0.75, cmap=ListedColormap(('red','green')))
  plt.title("SVM (Test Set)")
  plt.xlabel("Age")
  plt.ylabel("Estimated Salary")
  plt.legend()
  plt.show()
```

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that
```




```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that
```

SVM (Test Set)

✓ 10s completed at 11:42 PM