

```
#Import libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

```
#Load the Dataset
dataset = pd.read_csv('https://github.com/shivang98/Social-Network-ads-Boost/raw/master/Social_Network_Ads.csv')
```

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

```
#Split the X and Y Dataset into the Training set 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)
```

```
#Perform Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
#Fit SVM to the 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 Results
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)
accuracy_score(y_test, y_pred)
```

```
[[64  4]
 [ 3 29]]
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.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('SVM (Test set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

 <ipython-input-11-d61a781a61ac>:12: UserWarning: *c* argument looks like a single numeric RGB or RGBA sequence, which should be avo:
plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],

