

## ▼ Support Vector Machine (SVM)

### ▼ Importing the libraries

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

### ▼ Importing the dataset

```
dataset = pd.read_csv('Social_Network_Ads.csv')
X = dataset.iloc[:, -3:-1].values
y = dataset.iloc[:, -1].values
```

### ▼ Splitting the 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_

print(X_train)

print(y_train)

print(X_test)
```

The shortcut "Print notebook" is disabled when a code cell output iframe is active. Use the escape key to leave the iframe and enter the shortcut again. ✕

### ▼ Feature Scaling

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

print(X_train)

print(X_test)
```

## Training the SVM model on the Training set

### ▼ Predicting a new result

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

```
↳ SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
      decision_function_shape='ovr', degree=3, gamma='scale', kernel='linear',
      max_iter=-1, probability=False, random_state=0, shrinking=True, tol=0.001,
      verbose=False)
```

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

```
↳ [0]
```

### ▼ Predicting the Test set results

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

### ▼ Making the Confusion Matrix

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

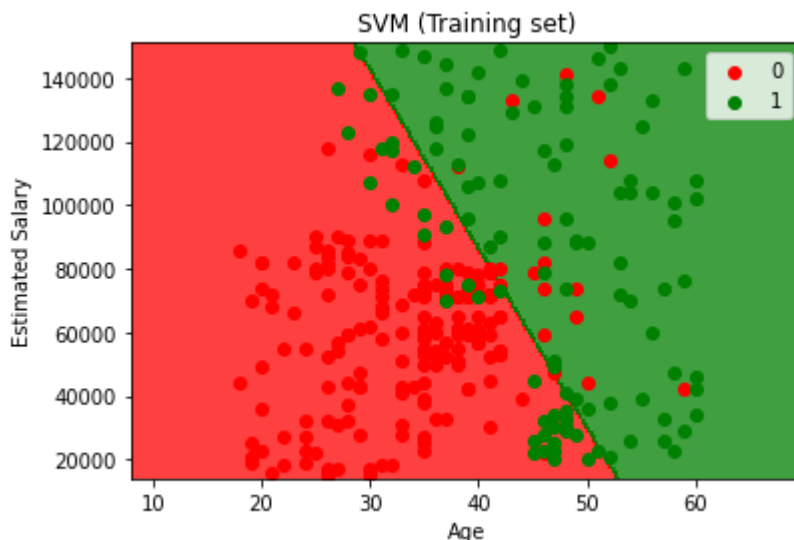
The shortcut "Print notebook" is disabled when a code cell output iframe is active. Use the escape key to leave the iframe and enter the shortcut again. ✕

### ▼ Visualising the Training set results

```
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, step = 5),
                     np.arange(start = X_set[:, 1].min() - 1000, stop = X_set[:, 1].max() + 1000, step = 100))
plt.contourf(X1, X2, classifier.predict(sc.transform(np.array([X1.ravel(), X2.ravel()])).reshape((-1, 2))),
             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],
```

```
plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1], c = ListedColormap(['red', 'green']))
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 :



## ▼ Visualising the Test set results

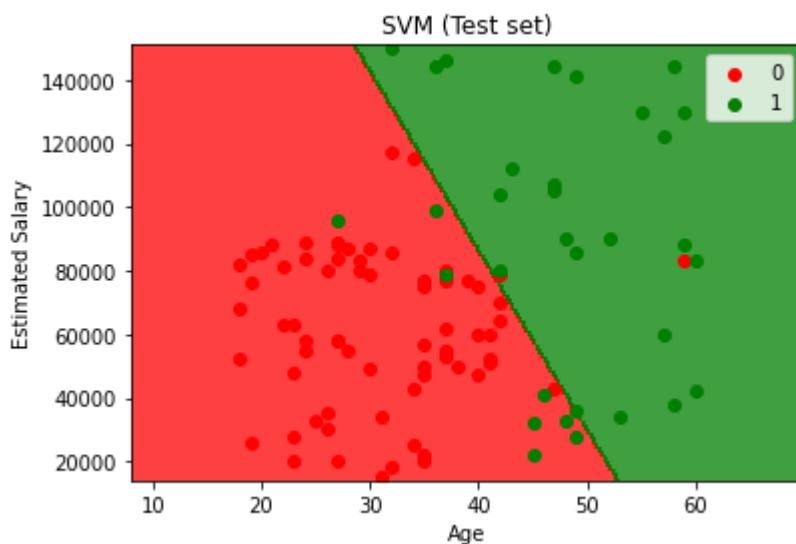
```
from matplotlib.colors import ListedColormap
X_set, y_set = sc.inverse_transform(X_test), y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 10, stop = X_set[:, 0].max() + 10, step = 5),
                     np.arange(start = X_set[:, 1].min() - 1000, stop = X_set[:, 1].max() + 1000, step = 1000))
plt.contourf(X1, X2, classifier.predict(sc.transform(np.array([X1.ravel(), X2.ravel()])).reshape((-1, 2))),
             alpha = 0.75, cmap = ListedColormap(['red', 'green']))
```

The shortcut "Print notebook" is disabled when a code cell output iframe is active. Use the escape key to leave the iframe and enter the shortcut again. ✕

```
plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1], c = ListedColormap(['red', 'green']))
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 :



The shortcut "Print notebook" is disabled when a code cell output iframe is active. Use the escape key to leave the iframe and enter the shortcut again.

