


```
# Support Vector Machine
# Importing the libraries

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



# Importing the datasets

datasets = pd.read_csv('Social_Network_Ads.csv')
X = datasets.iloc[:, [2,3]].values
Y = datasets.iloc[:, 4].values

datasets.head()
```



	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

Next steps:

[Generate code with datasets](#)[View recommended plots](#)[New interactive sheet](#)

```
# 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_state = 0)

# Feature Scaling

from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
X_Train = sc_X.fit_transform(X_Train)
X_Test = sc_X.transform(X_Test)

# Fitting the classifier into the Training set

from sklearn.svm import SVC
classifier = SVC(kernel = 'rbf')
classifier.fit(X_Train, Y_Train)
```



SVC

SVC()

```
# Predicting the test set results

Y_Pred = classifier.predict(X_Test)
```

```
from sklearn.metrics import confusion_matrix, accuracy_score
acc = accuracy_score(Y_Test, Y_Pred)
error_rate = 1 - (cm.diagonal().sum() / cm.sum())
print("Accuracy ", "{:.2%}".format(acc))
print("Error Rate:", error_rate)
```

```

→ Accuracy 93.00%
Error Rate: 0.14

```

```
# prompt: display accuracy and error rate in different lines and display accuracy in %
```

```

cm = confusion_matrix(Y_Test, Y_Pred)

acc=accuracy_score(Y_Test,Y_Pred)
error_rate = 1 - (cm.diagonal().sum() / cm.sum())
print("Accuracy:", "{:.2%}".format(acc))
print("Error Rate:", error_rate)

```

```

→ Accuracy: 93.00%
Error Rate: 0.06999999999999995

```

```

cm=confusion_matrix(Y_Test,Y_Pred)
cm

```

```

→ array([[64,  4],
        [ 3, 29]])

```

```

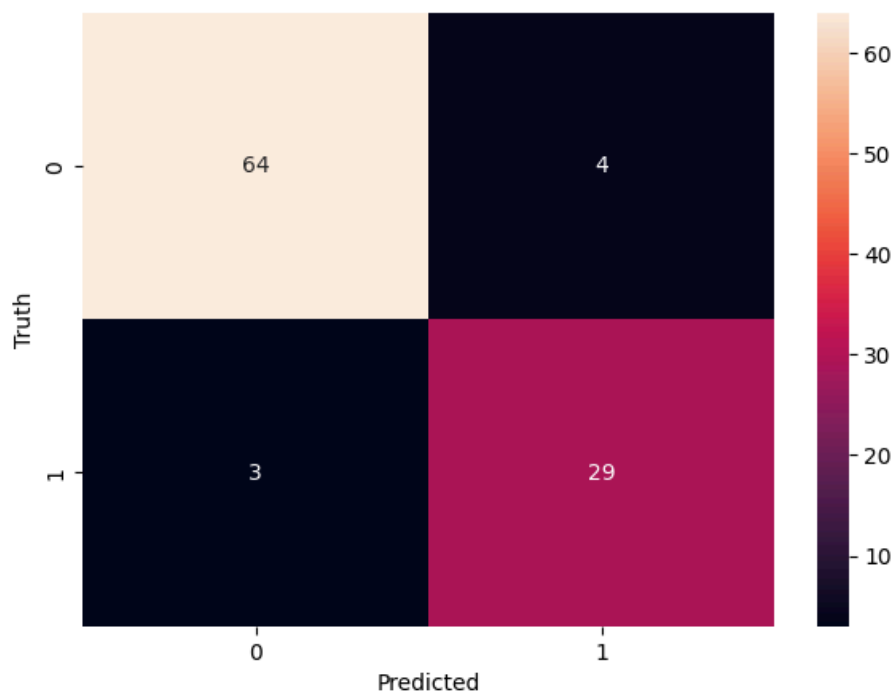
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(7, 5))
sns.heatmap(cm, annot=True)
plt.xlabel('Predicted')
plt.ylabel('Truth')

```

```

→ Text(58.22222222222214, 0.5, 'Truth')

```



```

from sklearn.metrics import classification_report

print(classification_report(Y_Test, Y_Pred))

```

```

→

```

	precision	recall	f1-score	support
0	0.96	0.94	0.95	68
1	0.88	0.91	0.89	32
accuracy			0.93	100

macro avg	0.92	0.92	0.92	100
weighted avg	0.93	0.93	0.93	100

```
# Visualising the Training set results
```

```
from matplotlib.colors import ListedColormap
X_Set, Y_Set = X_Train, Y_Train
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('Support Vector Machine (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

```
# Visualising 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('Support Vector Machine (Test set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

