Isha Sah
RA1911003010668

AI LAB-10

Aim: Implementation of learning algorithms for an application.

Problem Formulation: Solving a dataset using machine learning algorithms.

Problem Statement: Using a dataset to predict if a person in susceptible to heart attacks or not by taking values like blood pressure, cholesterol levels, etc.

Algorithm used (Problem Solving): Kernel SVM
The Support Vector Machine is a supervised learning algorithm mostly used for classification but it can be used also for regression. The main idea is that based on the labeled data (training data) the algorithm tries to find the optimal hyperplane which can be used to classify new data points. In two dimensions the hyperplane is a simple line. Kernel SVM deals with data having higher dimensions and non-linearity.

Dataset:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
2	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
3	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
4	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
5	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
6	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
7	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
8	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
9	44	1	1	120	263	0	1	173	0	0	2	0	3	1
10	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1

The dataset has 304 rows, i.e. 304 data entries. Output 1 means that the patient has more chances of having a heart attack and 0 means less chance.

Code:

Importing the libraries

In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

Importing the dataset

In [ ]:
```
dataset = pd.read_csv('heart.csv')
```

Data Preprocessing

In [ ]:
```
dataset.describe()
```

Out[ ]:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.399340	0.729373	2.313531	0.544554
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.616226	1.022606	0.612277	0.498835
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.000000	0.000000	2.000000	0.000000
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.000000	0.000000	2.000000	1.000000
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	3.000000	1.000000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000000	1.000000

In [ ]:
```
dataset.drop(['fbs', 'chol'], axis = 1, inplace = True)
```

In [ ]:
```
dataset
```

Out[ ]:

	age	sex	cp	trtbps	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
0	63	1	3	145	0	150	0	2.3	0	0	1	1
1	37	1	2	130	1	187	0	3.5	0	0	2	1
2	41	0	1	130	0	172	0	1.4	2	0	2	1
3	56	1	1	120	1	178	0	0.8	2	0	2	1
4	57	0	0	120	1	163	1	0.6	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	0	0	140	1	123	1	0.2	1	0	3	0
299	45	1	3	110	1	132	0	1.2	1	0	3	0
300	68	1	0	144	1	141	0	3.4	1	2	3	0
301	57	1	0	130	1	115	1	1.2	1	1	3	0
302	57	0	1	130	0	174	0	0.0	1	1	2	0

303 rows × 12 columns

In [ ]:
```
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
```

In [ ]:
```
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
```

In [ ]:
```
X
```

Out[ ]:
```
array([[63.,  1.,  3., ...,  0.,  0.,  1.],
       [37.,  1.,  2., ...,  0.,  0.,  2.],
       [41.,  0.,  1., ...,  2.,  0.,  2.],
       ...,
       [68.,  1.,  0., ...,  1.,  2.,  3.],
       [57.,  1.,  0., ...,  1.,  1.,  3.],
       [57.,  0.,  1., ...,  1.,  1.,  2.]])
```

Splitting the dataset into the Training set and Test set

In [ ]:
```
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

In [ ]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

Kernel SVM

In [ ]:
```
from sklearn.svm import SVC
classifier_svm = SVC(kernel = 'rbf', random_state = 0)
classifier_svm.fit(X_train, y_train)
y_pred_svm = classifier_svm.predict(X_test)
```

In [ ]:
```
cm_svm = confusion_matrix(y_test, y_pred_svm)
print(cm_svm)
accuracy_score(y_test, y_pred_svm)
```

Out[ ]:
```
[[24  9]
 [ 3 40]]
0.8421052631578947
```

Output: