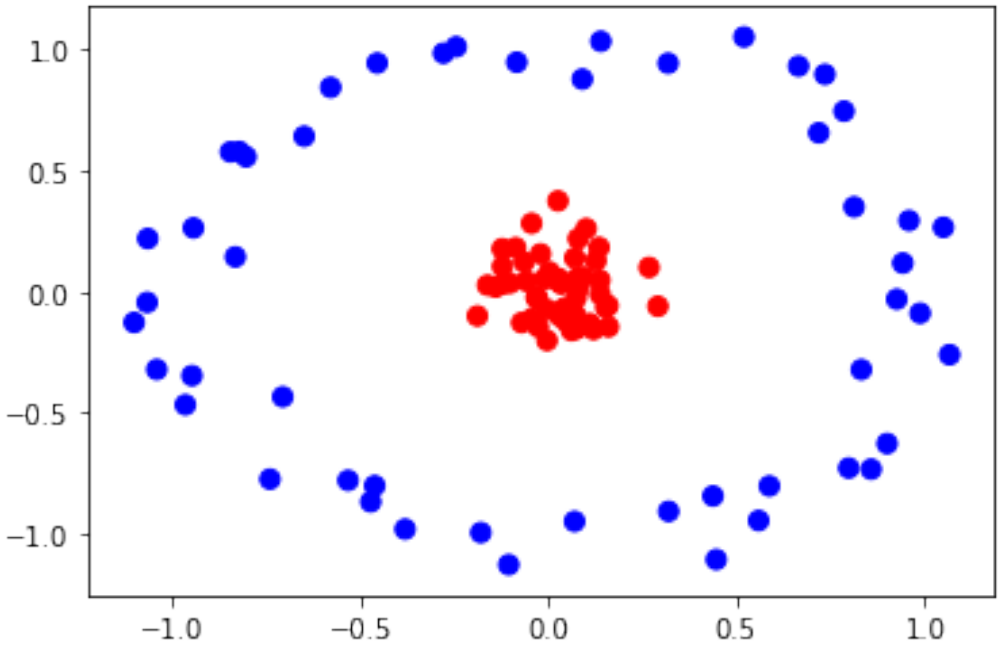
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.axes._axes import _log as matplotlib_axes_logger
from mpl_toolkits import mplot3d
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from matplotlib.colors import ListedColormap

from sklearn.datasets.samples_generator import make_circles
X, y = make_circles(100, factor=.1, noise=.1)

plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='bwr')
```

```
<matplotlib.collections.PathCollection at 0xc960607780>
```




```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
0.20)

classifier = SVC(kernel="linear")
classifier.fit(X_train, y_train.ravel())
y_pred = classifier.predict(X_test)

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

```
0.55
```

```
zero_one_colourmap = ListedColormap(('blue', 'red'))
def plot_decision_boundary(X, y, clf):
```

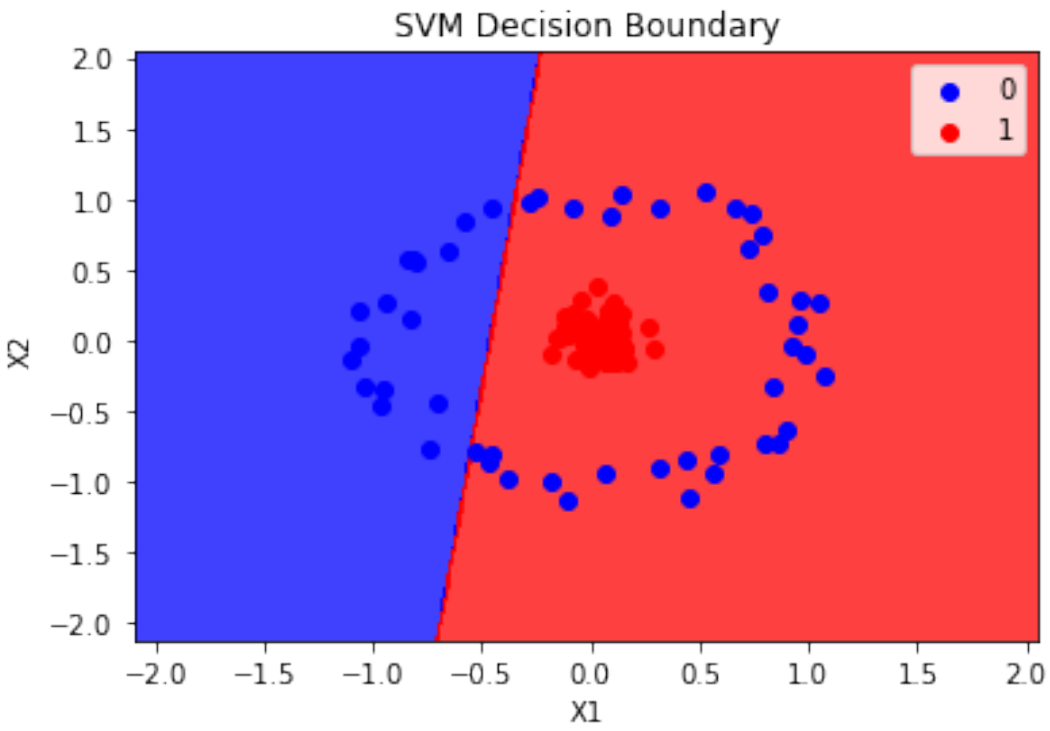
```
    X_set, y_set = X, y
    X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1,
                                   stop = X_set[:, 0].max() + 1,
                                   step = 0.01),
                         np.arange(start = X_set[:, 1].min() - 1,
                                   stop = X_set[:, 1].max() + 1,
                                   step = 0.01))

    plt.contourf(X1, X2, clf.predict(np.array([X1.ravel(),

X2.ravel()]).T).reshape(X1.shape),
                 alpha = 0.75,
                 cmap = zero_one_colourmap)
    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 = (zero_one_colourmap)(i), label = j)
    plt.title('SVM Decision Boundary')
    plt.xlabel('X1')
    plt.ylabel('X2')
    plt.legend()
    return plt.show()

plot_decision_boundary(X, y, classifier)
```

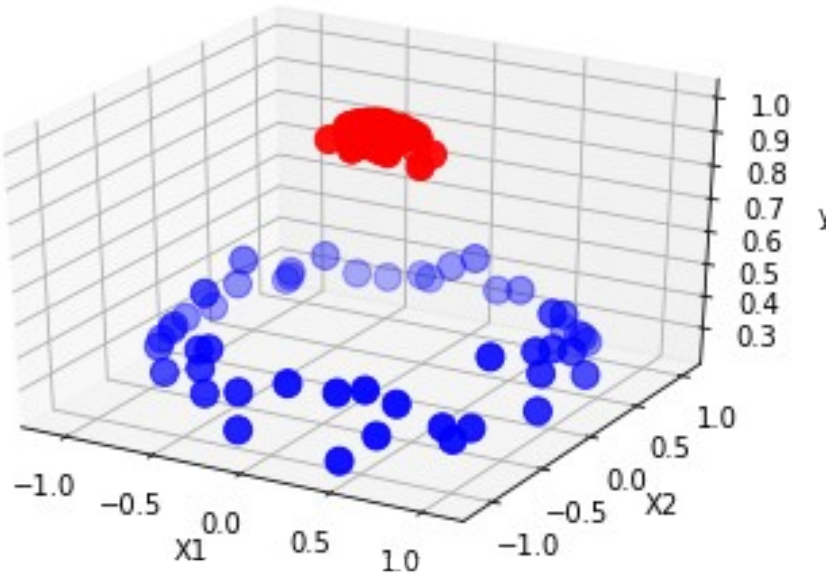
```
'c' argument looks like a single numeric RGB or RGBA sequence, which
should be avoided as value-mapping will have precedence in case its
length matches with 'x' & 'y'.  Please use a 2-D array with a single
row if you really want to specify the same RGB or RGBA value for all
points.
'c' argument looks like a single numeric RGB or RGBA sequence, which
should be avoided as value-mapping will have precedence in case its
length matches with 'x' & 'y'.  Please use a 2-D array with a single
row if you really want to specify the same RGB or RGBA value for all
points.
```




```
def plot_3d_plot(X, y):
    r = np.exp(-(X ** 2).sum(1))
    ax = plt.subplot(projection='3d')
    ax.scatter3D(X[:, 0], X[:, 1], r, c=y, s=100, cmap='bwr')
    ax.set_xlabel('X1')
    ax.set_ylabel('X2')
    ax.set_zlabel('y')
    return ax

plot_3d_plot(X,y)
```

```
<matplotlib.axes._subplots.Axes3DSubplot at 0xc9616bab00>
```




```
rbf_classifier = SVC(kernel="rbf")
rbf_classifier.fit(X_train, y_train)
y_pred = rbf_classifier.predict(X_test)
```

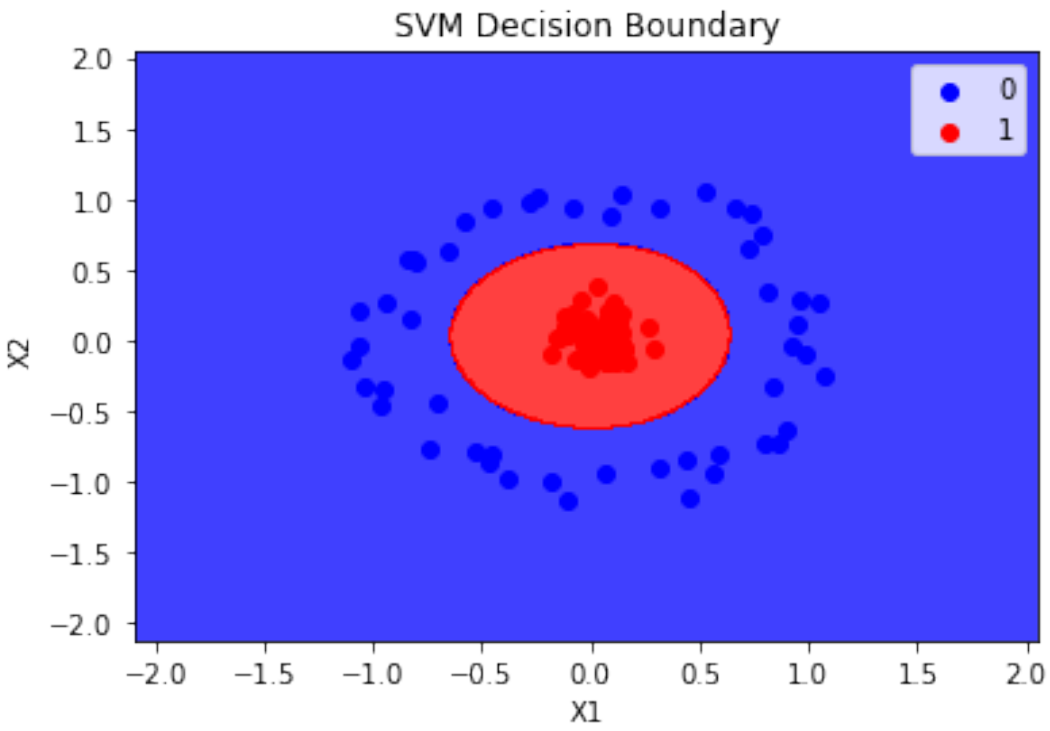
```
C:\Users\Nitish\Anaconda3\lib\site-packages\sklearn\svm\base.py:193:
FutureWarning: The default value of gamma will change from 'auto' to
'scale' in version 0.22 to account better for unscaled features. Set
gamma explicitly to 'auto' or 'scale' to avoid this warning.
  "avoid this warning.", FutureWarning)
```

```
accuracy_score(y_test, y_pred)
```

```
1.0
```

```
plot_decision_boundary(X, y, rbf_classifier)
```

```
'c' argument looks like a single numeric RGB or RGBA sequence, which
should be avoided as value-mapping will have precedence in case its
length matches with 'x' & 'y'.  Please use a 2-D array with a single
row if you really want to specify the same RGB or RGBA value for all
points.
'c' argument looks like a single numeric RGB or RGBA sequence, which
should be avoided as value-mapping will have precedence in case its
length matches with 'x' & 'y'.  Please use a 2-D array with a single
row if you really want to specify the same RGB or RGBA value for all
points.
```




```
poly_classifier = SVC(kernel="poly",degree=2)
poly_classifier.fit(X_train, y_train)
y_pred = poly_classifier.predict(X_test)
```

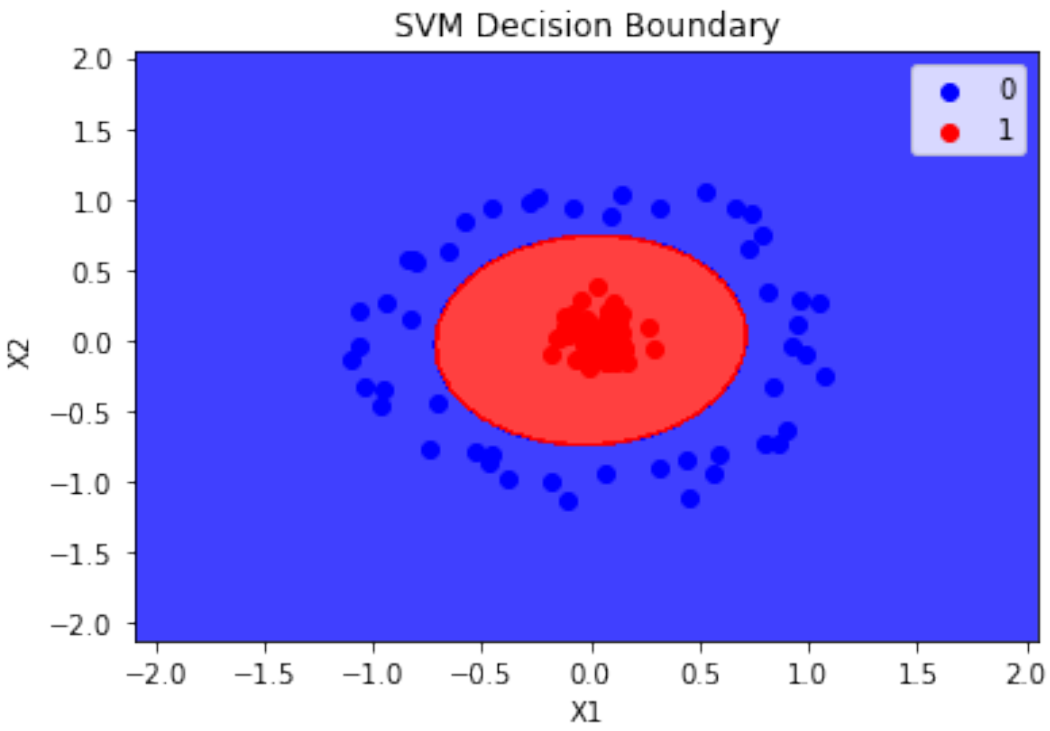
```
C:\Users\Nitish\Anaconda3\lib\site-packages\sklearn\svm\base.py:193:
FutureWarning: The default value of gamma will change from 'auto' to
'scale' in version 0.22 to account better for unscaled features. Set
gamma explicitly to 'auto' or 'scale' to avoid this warning.
  "avoid this warning.", FutureWarning)
```

```
accuracy_score(y_test, y_pred)
```

```
1.0
```

```
plot_decision_boundary(X, y, poly_classifier)
```

```
'c' argument looks like a single numeric RGB or RGBA sequence, which
should be avoided as value-mapping will have precedence in case its
length matches with 'x' & 'y'.  Please use a 2-D array with a single
row if you really want to specify the same RGB or RGBA value for all
points.
'c' argument looks like a single numeric RGB or RGBA sequence, which
should be avoided as value-mapping will have precedence in case its
length matches with 'x' & 'y'.  Please use a 2-D array with a single
row if you really want to specify the same RGB or RGBA value for all
points.
```



X

```
array([[ 0.39896842,  1.16747207],
       [ 0.36174994,  0.16028343],
       [-0.01280271, -0.1227449 ],
       [-0.91697399,  0.28284427],
       [ 0.10959615, -0.89264008],
       [ 0.23023935,  0.02235324],
       [-0.04124462, -0.02506236],
       [-0.04227921, -0.20651006],
       [ 0.35957795,  0.95449362],
       [ 0.82413796, -0.50526127],
       [ 0.98105034, -0.14023386],
       [ 0.88251666,  0.5295689 ],
       [ 0.03693396, -0.10528597],
       [-0.51177477, -0.78506493],
       [ 0.04697092, -0.23392657],
       [-0.08206585, -0.04615685],
       [ 0.0359025 ,  0.09810491],
       [ 0.26770521,  0.28702445],
       [-1.02190094,  0.01891189],
       [ 0.66685164,  0.62375836],
       [-0.16285115, -0.31172126],
       [-0.21602558, -0.02210426],
       [-0.02330554,  0.92783582],
       [-0.07078896,  0.82481082],
       [ 0.00344906, -0.05740011],
```

```
       [-0.02202348,  0.14869388],
       [-0.22084022,  1.0254179 ],
       [-0.19711251,  0.98327735],
       [-0.51437148,  0.94991734],
       [ 0.57859321,  0.88650324],
       [ 1.15066913, -0.34633592],
       [ 0.01445866,  0.03308021],
       [ 0.81069878,  0.64403127],
       [ 0.09317087, -0.1829618 ],
       [-0.7453608 ,  0.63432868],
       [ 0.7372497 ,  0.48959721],
       [ 0.00221705, -0.90246911],
       [ 0.27145256,  0.94843314],
       [-0.43166027, -0.9301172 ],
       [ 0.11151432,  0.03803949],
       [-0.66946093,  0.36388581],
       [-0.08917277, -0.02904766],
       [-1.01282923, -0.34285532],
       [ 0.58609773,  0.94133312],
       [ 0.03862903, -0.10450585],
       [-0.08335726,  0.09578953],
       [ 0.02938388, -0.02380587],
       [-0.02311158, -0.19596673],
       [-0.63455833,  0.67128113],
       [ 0.8901117 , -0.61613816],
       [-1.07486834,  0.150548  ],
       [ 1.10423365,  0.08375884],
       [-0.09384108, -0.16866621],
       [ 0.11687845, -0.16979057],
       [ 0.04286886,  0.14282573],
       [-0.18255853, -0.91669436],
       [ 0.13205677, -0.95399843],
       [-0.12022273,  0.09468948],
       [-0.02185074, -0.02638744],
       [ 0.65638326, -0.70239899],
       [-0.90540201,  0.45426359],
       [-0.1039018 ,  0.03982494],
       [ 0.04839703, -0.03304355],
       [ 0.22313027, -0.09382891],
       [ 1.0803262 ,  0.37081246],
       [ 0.40757845, -0.83242259],
       [-0.25059408,  0.21435611],
       [ 1.06086631, -0.05866112],
       [-0.49387227,  0.70185683],
       [ 0.16236529, -0.09226957],
       [ 0.07560523,  0.09107179],
       [-0.07611597,  0.099193  ],
       [ 0.02530438,  0.12909692],
       [-0.02940615, -0.06206698],
```

```
       [ 0.85299145, -0.63979064],
       [-0.02956193,  0.26031196],
       [ 0.03978148,  0.07960912],
       [ 0.08451198, -0.06991151],
       [-0.33007214, -0.10134742],
       [-0.85479344,  0.54903121],
       [-0.18042997,  0.04491498],
       [ 0.07048815,  0.1457222 ],
       [ 0.01080809,  0.18150986],
       [ 1.04578516, -0.10975758],
       [ 0.21445745,  0.07714541],
       [-0.65420902, -0.75615957],
       [-0.15890112,  0.04862769],
       [-0.95986862, -0.31929751],
       [-0.8157016 , -0.44529218],
       [ 0.35282391, -0.98520706],
       [ 0.01835522, -0.19625642],
       [-0.35536673, -1.0295979 ],
       [ 0.63216274, -0.73995001],
       [-0.70179134, -0.57760657],
       [-0.06470009,  0.13994144],
       [ 0.01484427,  0.19953292],
       [-0.76782085, -0.3008828 ],
       [-0.84944705, -0.61397925],
       [ 0.21070272, -0.0454922 ],
       [ 0.05948227, -0.12663255]])
```

```
np.exp(-(X**2)).sum(1)
```

```
array([1.92474932, 1.30551477, 1.33451789, 1.11434663, 1.86888988,
       1.98209455, 1.27142264, 1.98017671, 1.99475996, 1.99266903,
       1.28108279, 1.30240161, 1.99110906, 1.96821791, 1.31703694,
       1.45586193, 1.95382544, 1.9217843 , 1.11942537, 1.9864715 ,
       1.25504359, 1.99837501, 1.1979308 , 1.22525191, 1.34254909,
       1.98098055, 1.94943117, 1.98110864, 1.11397757, 1.99324997,
       1.36864613, 1.97056757, 1.14157194, 1.11728861, 1.40161855,
       1.98398902, 1.20677455, 1.97189796, 1.99367818, 1.06387675,
       1.24779089, 1.98773366, 1.34377658, 1.99739375, 1.32529426,
       1.99394329, 1.96003133, 1.29386874, 1.97858885, 1.99342189,
       1.97290436, 1.40233646, 1.97609461, 1.97574809, 1.99639779,
       1.43533448, 1.9975871 , 1.2242329 , 1.98872888, 1.96023923,
       1.99836278, 1.9471979 , 1.31844817, 1.96147996, 1.97112125,
       1.2400487 , 1.2699872 , 1.95487084, 1.97444787, 1.06373879,
       1.91515465, 1.47988897, 1.40218898, 1.99439689, 1.26409582,
       1.97810989, 1.92108766, 1.12760079, 1.03072363, 1.27481893,
       1.25810824, 1.20725387, 1.42314623, 1.4018611 , 1.31957482,
       1.09514099, 1.98567075, 1.99783889, 1.31680937, 1.34044122,
       1.9883814 , 1.23389513, 1.95425384, 1.97928021, 1.39785869,
       1.31559584, 1.24782266, 1.97339383, 1.29813936, 1.97065552])
```

```
X_new=np.exp(-(X**2))
plt.scatter(X_new[:, 0], X_new[:, 1], c=y, s=50, cmap='bwr')
```

```
<matplotlib.collections.PathCollection at 0xc961a9b470>
```