

Lab Assignment 6

Fundamentals of Machine Learning

Name: M Jeyapathy

USN: 22BTRAD016

1. Implementing SVM algorithm

```
# Import necessary libraries

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import make_classification

from sklearn.svm import SVC


# Create a small synthetic dataset

X, y = make_classification(n_samples=100, n_features=2, n_classes=2,
n_clusters_per_class=1, n_redundant=0, random_state=42)


# Create an SVM classifier

svm_classifier = SVC(kernel='linear')

svm_classifier.fit(X, y)


# Visualize the decision boundary

plt.figure(figsize=(8, 6))


# Plot data points

plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Paired, edgecolors='k', marker='o', s=100)


# Plot the decision boundary

ax = plt.gca()

xlim = ax.get_xlim()
```

```
ylim = ax.get_ylim()
```

```
# Create grid to evaluate model
```

```
xx, yy = np.meshgrid(np.linspace(xlim[0], xlim[1], 100), np.linspace(ylim[0], ylim[1], 100))
```

```
Z = svm_classifier.decision_function(np.c_[xx.ravel(), yy.ravel()])
```

```
# Plot decision boundary and margins
```

```
Z = Z.reshape(xx.shape)
```

```
plt.contour(xx, yy, Z, colors='k', levels=[-1, 0, 1], alpha=0.5, linestyles=['--', '-', '--'])
```

```
# Highlight the support vectors
```

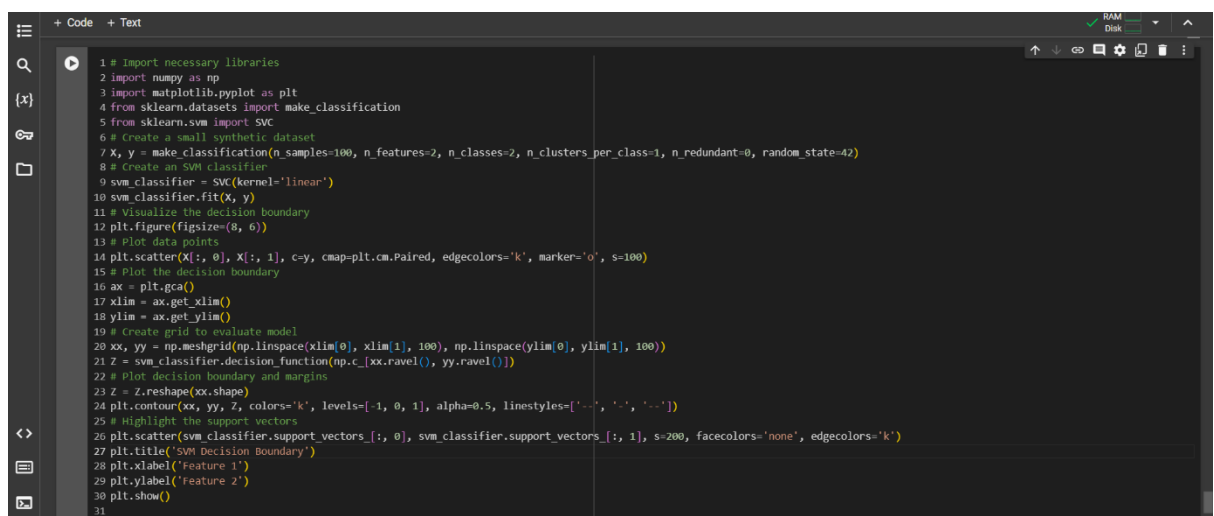
```
plt.scatter(svm_classifier.support_vectors_[:, 0], svm_classifier.support_vectors_[:, 1],  
s=200, facecolors='none', edgecolors='k')
```

```
plt.title('SVM Decision Boundary')
```

```
plt.xlabel('Feature 1')
```

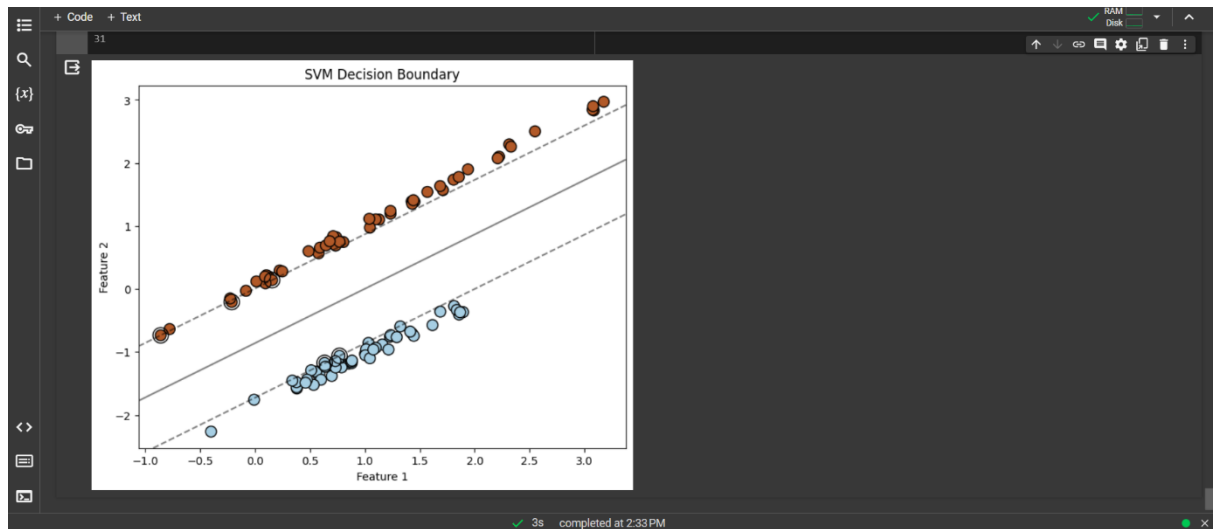
```
plt.ylabel('Feature 2')
```

```
plt.show()
```



```
+ Code + Text
1 # Import necessary libraries
2 import numpy as np
3 import matplotlib.pyplot as plt
4 from sklearn.datasets import make_classification
5 from sklearn.svm import SVC
6 # Create a small synthetic dataset
7 X, y = make_classification(n_samples=100, n_features=2, n_classes=2, n_clusters_per_class=1, n_redundant=0, random_state=42)
8 # Create an SVM classifier
9 svm_classifier = SVC(kernel='linear')
10 svm_classifier.fit(X, y)
11 # Visualize the decision boundary
12 plt.figure(figsize=(8, 6))
13 # Plot data points
14 plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Paired, edgecolors='k', marker='o', s=100)
15 # Plot the decision boundary
16 ax = plt.gca()
17 xlim = ax.get_xlim()
18 ylim = ax.get_ylim()
19 # Create grid to evaluate model
20 xx, yy = np.meshgrid(np.linspace(xlim[0], xlim[1], 100), np.linspace(ylim[0], ylim[1], 100))
21 Z = svm_classifier.decision_function(np.c_[xx.ravel(), yy.ravel()])
22 # Plot decision boundary and margins
23 Z = Z.reshape(xx.shape)
24 plt.contour(xx, yy, Z, colors='k', levels=[-1, 0, 1], alpha=0.5, linestyles=['--', '-', '--'])
25 # Highlight the support vectors
26 plt.scatter(svm_classifier.support_vectors_[:, 0], svm_classifier.support_vectors_[:, 1], s=200, facecolors='none', edgecolors='k')
27 plt.title('SVM Decision Boundary')
28 plt.xlabel('Feature 1')
29 plt.ylabel('Feature 2')
30 plt.show()
31
```

Output:



Github Link:

<https://github.com/Jeyapathy/Machine-Learning>