SVM

January 10, 2022

[1]:
```
# Import the modules
import sklearn
from sklearn import datasets
from sklearn import svm
from sklearn import metrics
```

[2]:
```
# Load a dataset from sklearn
cancer = datasets.load_breast_cancer()
```

[3]:
```
# Check the feature names of the dataset
print(cancer.feature_names)
```

```
['mean radius' 'mean texture' 'mean perimeter' 'mean area'
 'mean smoothness' 'mean compactness' 'mean concavity'
 'mean concave points' 'mean symmetry' 'mean fractal dimension'
 'radius error' 'texture error' 'perimeter error' 'area error'
 'smoothness error' 'compactness error' 'concavity error'
 'concave points error' 'symmetry error' 'fractal dimension error'
 'worst radius' 'worst texture' 'worst perimeter' 'worst area'
 'worst smoothness' 'worst compactness' 'worst concavity'
 'worst concave points' 'worst symmetry' 'worst fractal dimension']
```

[4]:
```
# Check the target names of the dataset
print(cancer.target_names)
```

```
['malignant' 'benign']
```

[5]:
```
# Define training and testing data
x = cancer.data
y = cancer.target
```

[6]:
```
# Split the data into train and test subsets
x_train, x_test, y_train, y_test = sklearn.model_selection.train_test_split(x,␣
 ↪y, test_size=0.2)
```

[7]:
```
# Check the training data
# 0 represents 'malignant'
# 1 represents 'benign'
```

```
print(x_train, y_train)
```

```
[[1.219e+01 1.329e+01 7.908e+01 … 8.187e-02 3.469e-01 9.241e-02]
 [1.246e+01 2.404e+01 8.397e+01 … 2.210e-01 4.366e-01 2.075e-01]
 [1.364e+01 1.634e+01 8.721e+01 … 8.586e-02 2.346e-01 8.025e-02]
 …
 [1.289e+01 1.570e+01 8.408e+01 … 1.017e-01 1.999e-01 7.127e-02]
 [1.088e+01 1.562e+01 7.041e+01 … 7.966e-02 2.581e-01 1.080e-01]
 [1.426e+01 1.965e+01 9.783e+01 … 1.505e-01 2.398e-01 1.082e-01]] [1 0 1 1 0 0
1 0 0 0 0 1 0 1 1 1 1 0 0 1 0 1 1 1 1 1 0 1 1 1 0 1 1 1 1 0 0
 0 0 1 1 1 0 0 1 1 1 1 0 0 1 1 0 1 0 1 1 1 0 0 1 1 1 1 0 1 1 1 1 1 1 1 0
 1 1 1 0 0 1 1 1 1 1 0 1 0 1 1 1 0 1 0 0 1 0 0 0 1 1 1 1 0 1 1 0 0 0 1 0 1
 0 0 1 1 0 0 0 1 1 1 1 1 0 1 1 1 0 1 0 1 0 0 1 0 1 1 1 1 0 1 0 1 1 1 1 0 1
 0 1 1 1 0 0 1 1 0 1 1 1 1 0 1 1 0 1 1 1 0 1 0 1 1 1 1 0 1 1 1 0 1 0 1 1 1
 0 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 0 1 0 0 0 1 0 1 1 1 1 0 1 0 1 1
 0 1 1 1 1 1 0 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0
 1 1 1 1 1 1 1 1 1 0 0 1 1 0 0 0 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 1 0 1 0 1
 1 1 1 1 0 1 0 1 1 1 1 0 1 1 1 0 0 0 0 0 1 1 1 0 0 0 0 1 0 0 1 0 1 1 0 0 0
 1 1 0 0 1 1 1 1 1 1 1 0 1 1 0 0 0 1 1 0 0 0 1 1 0 0 1 0 0 0 0 1 1 0 1 1 1
 1 0 1 0 0 1 1 0 1 0 1 0 1 1 1 1 0 0 1 0 0 0 1 1 1 0 1 1 1 1 1 1 0 0 1 0 1
 1 0 1 1 0 1 1 1 1 1 1 1 0 1 1 0 1 1 0 1 0 1 1 1 1 0 0 1 1 0 1 1 0 0 1 1 1
 1 1 1 1 0 1 0 1 1 1 1]
```

[8]:
```
# Turn the integers into 'malignant' or 'benign'
classes = ['malignant' 'benign']
```

[9]:
```
# Make a Support Vector Machine by applying a Support Vector Classifier
# Apply a linear kernel
# Apply a soft margin of 1
clf = svm.SVC(kernel="linear", C=1)
```

[10]:
```
# Fit the Support Vector Machine
clf.fit(x_train, y_train)
```

[10]:
```
SVC(C=1, kernel='linear')
```

[11]:
```
# Make predictions based on the test data
y_pred = clf.predict(x_test)
```

[12]:
```
acc = metrics.accuracy_score(y_test, y_pred)
```

[13]:
```
print(acc)
```

```
0.9736842105263158
```

With a support vector classifier with a kernel and soft margin of 1. The result of the accuracy of the svm model is 99%

[ ]: