# Support Vector Machines in sklearn

In this section, you'll use support vector machines to fit a given sample dataset.

Before you do that, let's go over the tools required to build this model.

For your support vector machine model, you'll be using scikit-learn's `SVC` class. This class provides the functions to define and fit the model to your data.

```
>>> from sklearn.svm import SVC
>>> model = SVC()
>>> model.fit(x_values, y_values)
```

In the example above, the `model` variable is a support vector machine model that has been fitted to the data `x_values` and `y_values`. Fitting the model means finding the best boundary that fits the training data. Let's make two predictions using the model's `predict()` function.

```
>>> print(model.predict([ [0.2, 0.8], [0.5, 0.4] ]))
[[ 0., 1.]]
```

The model returned an array of predictions, one prediction for each input array. The first input, `[0.2, 0.8]`, got a prediction of `0.`. The second input, `[0.5, 0.4]`, got a prediction of `1.`.

## Hyperparameters

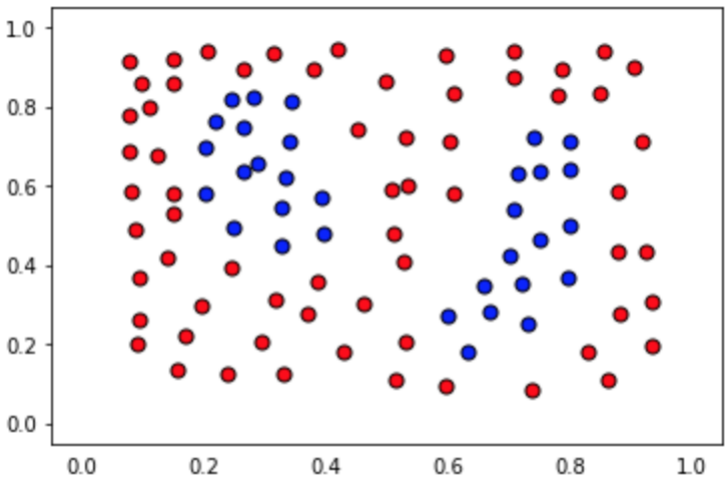
When we define the model, we can specify the hyperparameters. As we've seen in this section, the most common ones are

- `C`: The C parameter.
- `kernel`: The kernel. The most common ones are 'linear', 'poly', and 'rbf'.
- `degree`: If the kernel is polynomial, this is the maximum degree of the monomials in the kernel.
- `gamma`: If the kernel is rbf, this is the gamma parameter.

```
>>> model = SVC(kernel='poly', degree=4, C=0.1)
```

## Support Vector Machines Quiz

In this quiz, you'll be given with the following sample dataset, and your goal is to define a model that gives 100% accuracy on it.




The data file can be found under the "data.csv" tab in the quiz below. It includes three columns, the first 2 comprising of the coordinates of the points, and the third one of the label.

The data will be loaded for you, and split into features `X` and labels `y`.

### You'll need to complete each of the following steps:

**1. Build a support vector machine model**

- Create a support vector machine classification model using scikit-learn's **`SVC`** and assign it to the variable `model`.

**2. Fit the model to the data**

- If necessary, specify some of the hyperparameters. The goal is to obtain an accuracy of 100% in the dataset. *Hint: Not every kernel will work well.*

SVMs in sklearn

- Predict the labels for the training set, and assign this list to the variable `y_pred`.

### 4. Calculate the accuracy of the model

- For this, use the function sklearn function `accuracy_score`.

When you hit **Test Run**, you'll be able to see the boundary region of your model, which will help you tune the correct parameters, in case you need them.

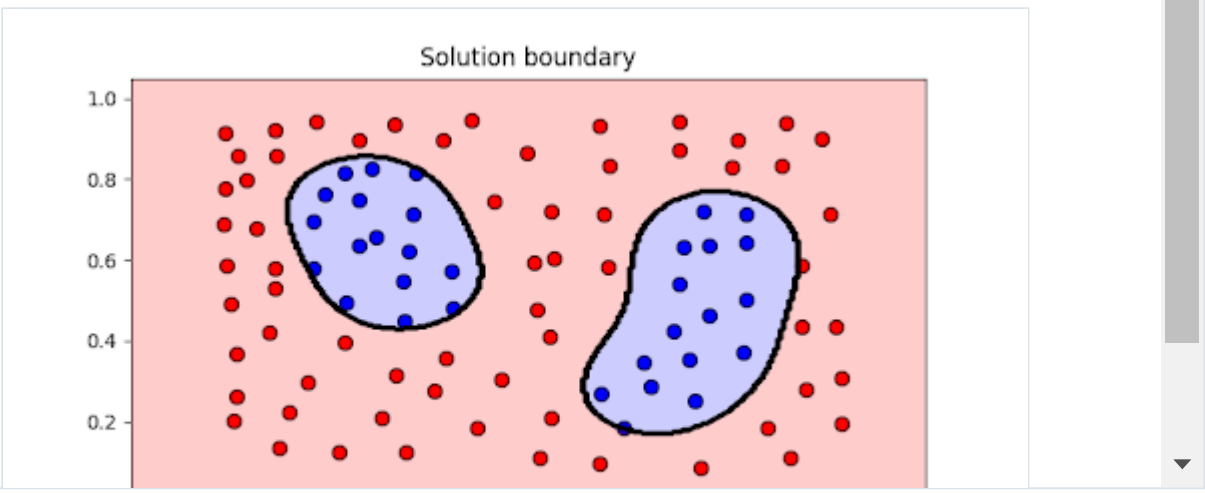
***Note:*** This quiz requires you to find an accuracy of 100% on the training set. Of course, this screams overfitting! If you pick very large values for your parameters, you will fit the training set very well, but it may not be the best model. Try to find the smallest possible parameters that do the job, which has less chance of overfitting, although this part won't be graded.

quiz.py | data.csv | solution.py

```
# Import statements
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
import pandas as pd
import numpy as np

# Read the data.
data = np.asarray(pd.read_csv('data.csv', header=None))
# Assign the features to the variable X, and the labels to the variable y.
X = data[:,0:2]
y = data[:,2]

# TODO: Create the model and assign it to the variable model.
# Find the right parameters for this model to achieve 100% accuracy on the datas
model = SVC(kernel ='rbf', gamma= 27)

# TODO: Fit the model.

model = model.fit(X,y)

# TODO: Make predictions. Store them in the variable y_pred.

y_pred = model.predict(X)

# TODO: Calculate the accuracy and assign it to the variable acc.
acc = accuracy_score(y_pred, y)
```

SVMs in sklearn

```
Great job!
```




RESET QUIZ TEST RUN SUBMIT ANSWER

NEXT