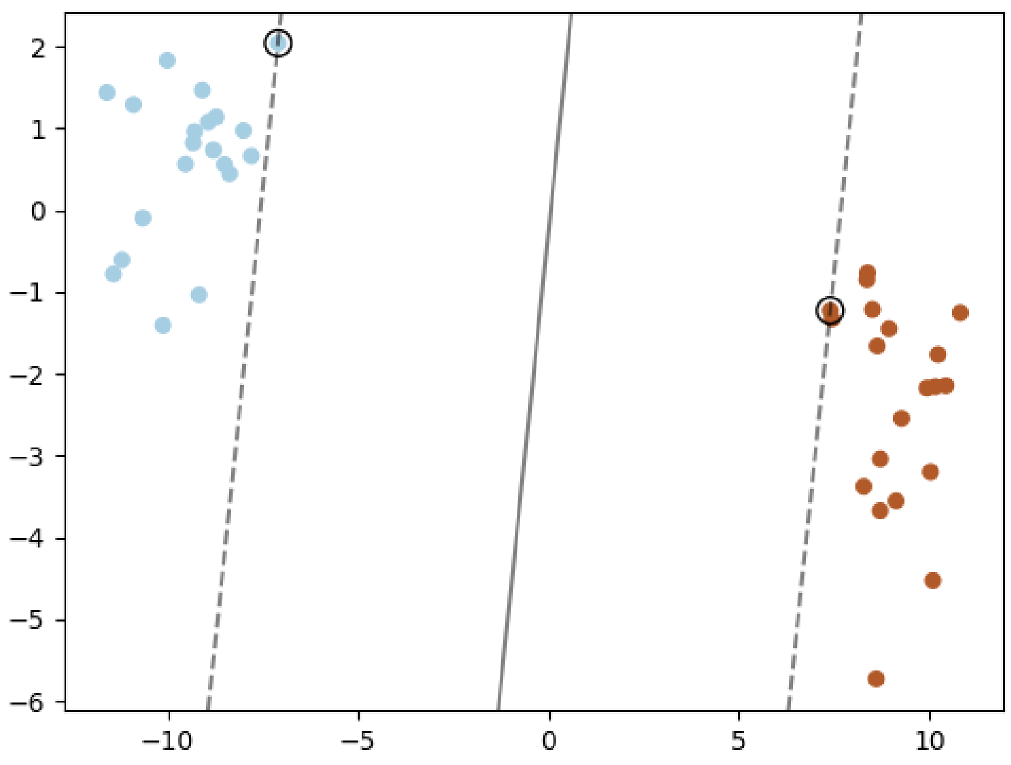
Sigurður Ágúst Jakobsson

# Project: Assignment 8 - Support Vector Machines

## Section 1.1

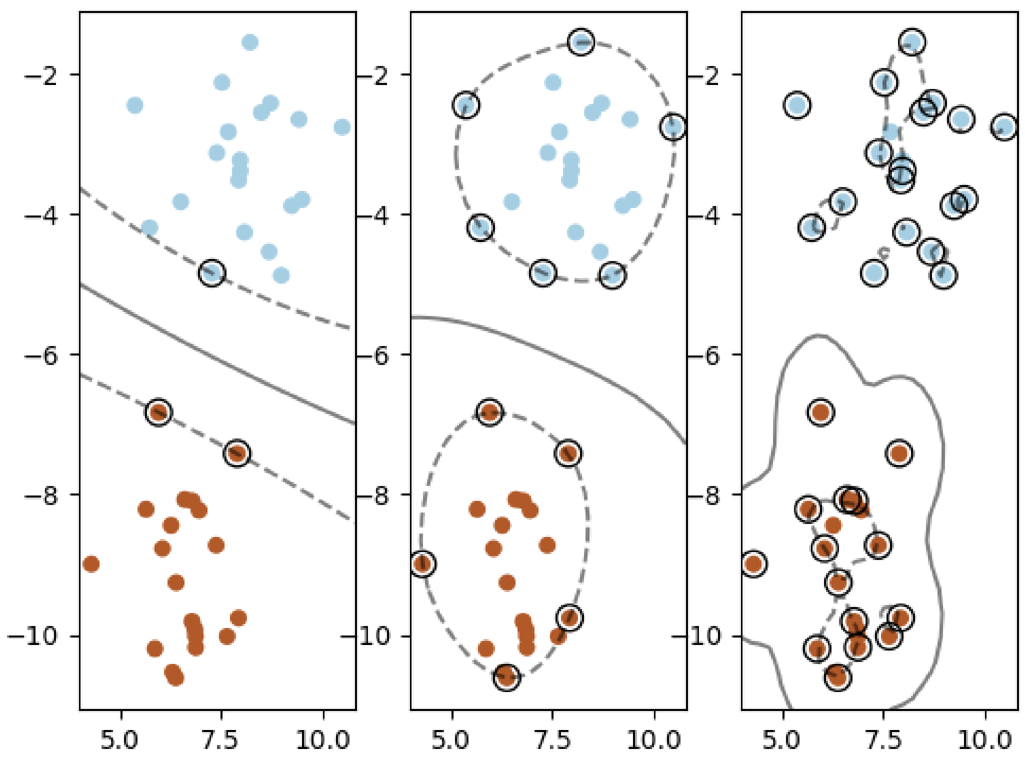


## Section 1.2

1. How many support vectors are there for each class in your example?
2. What is the shape of the decision boundary?

   There is one support vector for each class in this example and the shape of the decision boundary is a straight line.  This makes sense since we are using a linear kernel in the SVM.

## Section 1.3



## Section 1.4

1. How many support vectors are there for each class for each value of `gamma`?
2. What is the shape of the decision boundary for each value of `gamma`?
3. What difference does the `gamma` parameter make and why?

In this example the first class is blue and the second brown.

The support vector count is as follows using the .n_support_ attribute from sklearn:

Number of SVs Gamma-default: [1 2]

Number of SVs Gamma-0.2: [6 5]

Number of SVs Gamma-2.0: [18 15]

The shape of the decision boundary for each value of gamma is:

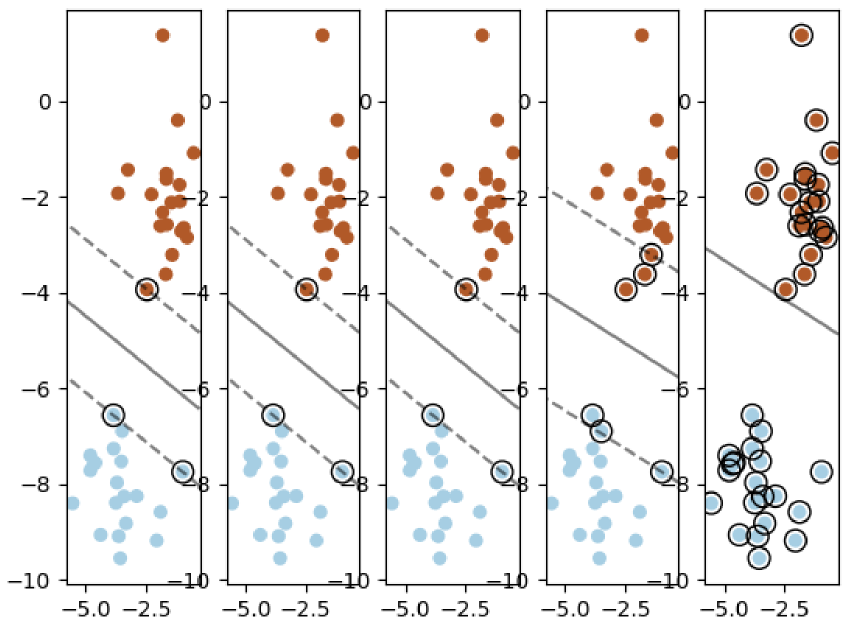
Gamma-default: Curved line

Gamma-0.2:  More curved line

Gamma-2.0: Curved enclosure

The sklearn documentation describes gamma as defining *"how far the influence of a single training example reaches"*. The higher that gamma is the closer this influence is. I assume that the default value is calculated to be under 0.2 since they define it as *1 / (n_features * X.var())*. This makes the influence of class two smaller as we increase gamma I assume. We get a smaller and smaller enclosure for these points, until they are enclosed in a small decision space.

## Section 1.5



## Section 1.6

1. How many support vectors are there for each class for each case of C?
2. How many of those support vectors are within the margins?.
3. Are any support vectors misclassified? If so, why?

In this example the first class is blue and the second brown.

The support vector count is as follows using the .n_support_ attribute from sklearn:

Number of SVs C-1000: [2 1]

Number of SVs C-0.5: [2 1]

Number of SVs C-0.3: [2 1]

Number of SVs C-0.05: [3 3]

Number of SVs C-0.0001: [20 20]

Regarding how many of the support vectors are within the margins I have to look at this visually. I count ones that are completely within the margin, but I assume there could be a little difference in reality, if the centres of the circles are slightly within the margin.

Number of SVs within margin C-1000: [0 0]

Number of SVs within margin C-0.5: [0 0]

Number of SVs within margin C-0.3: [0 0]

Number of SVs within margin C-0.05: [1 2]

Number of SVs within margin C-0.0001: [20 20]

No support vectors are misclassified. I assume the classification is good since the data has a good split for a linear classifier and there are no significant outliers.

## Section 2.2

Since the data generation is somewhat random I ran the comparison 100 times and did a 95% two tailed confidence test on the results. This takes about 25 minutes to run on this setup. That includes 300 fittings of the data. The data is split into [min, mean, max].

A linear classifier had the highest accuracy, radial the second highest and polynomial the lowest on average. They are still all within the same intervals and relatively close to each other.

Linear accuracy 95% [0.9261222459876465, 0.954736842105263, 0.9833514382228795]

Linear precision 95% [0.9244939945832857, 0.9619909247582945, 0.9994878549333033]

Linear recall 95% [0.9299468585682926, 0.9667232254036431, 1.0034995922389935]

RBF accuracy 95% [0.9246491540959695, 0.949766081871345, 0.9748830096467206]

RBF precision 95% [0.9141815541702972, 0.9503008679481928, 0.9864201817260884]

RBF recall 95% [0.9442835294767578, 0.9712425183192093, 0.9982015071616608]

Poly accuracy 95% [0.8985045734312762, 0.9322807017543857, 0.9660568300774952]

Poly precision 95% [0.8754561898541681, 0.9252664545750217, 0.9750767192958752]

Poly recall 95% [0.9431909944584462, 0.9713222958053481, 0.99945359715225]

We might not want to optimize accuracy though when looking for breast cancer. A reasonable goal would be to want to minimize false negatives, so we don't miss patients that have cancer. This is probably the costliest mistake to make in this context. Then the goal would be to maximize recall which is defined as TP/(TP+FN). While the polynomial kernel function had the lowest accuracy on average, it has the highest recall, slightly edging out the radial basis kernel. I would therefore argue that it would be best to use the polynomial kernel function for this task. My first instinct was to say that the RBF would be a good medium since it is only about 0,01% lower. On the scale of a million tests this small difference would mean 100 missed diagnoses, however. I think one should absolutely minimize these false negatives and use the kernel function with the highest recall by any margin.

## Independent Section

N/A – Don’t have an interesting idea to test out within my time budget this time.