Kernel methods

We can do non-linear classification by explicitly creating a new features. The challenge here is that we don't really know what new feature space to use?

So for example in the slides we see that (x_1, x_2) => $({x_1}^2, {x_2}^2)$ worked well for the circle separator. But that is because we can see the data visually. In general we have more than two dimensions and so what is the best transformation?

Consider the transformation into all powers up to 2.

So this means $\Phi(x_1)$ => (1, x_1, ${x_1}^2$)

$$\Phi(x_1, x_2) \; = \; (1, x_1, x_2, x_1x_2, {x_1}^2, {x_2}^2)$$

$$\Phi(x_1, x_2, x_3) \; = \; (1, x_1, x_2, x_3, x_1x_2, x_1x_3, x_2x_3, {x_1}^2, {x_2}^2, {x_3}^2)$$

$\Phi(x)$is the polynomial degree 2 kernel transformation.

If the original feature space has d dimensions then how many in the new one with powers up to 2?

We have 1 + d + d + (d choose 2) which is order of d^2.

What about powers of three? The new feature space would be order of d^3.

This explicit method is a very powerful method because if we know something about our underlying data then we can come up with powerful non-linear classifiers. For example suppose you are told to build a classifier with this property:

Data is three dimensional (x_1, x_2, x_3)

f(x) = -1 if $a * e^{x_1} + b * log(x_2) \; + \; c * x_2 * x_3 < \; 0$
+1 otherwise

The right hand side of the above is a dot product between w=(a,b,c) and $(e^{x_1}, log(x_2), x_2 * x_3)$

We can easily perform the transformation (x1, x2, x3) =>$(e^{x_1}, log(x_2), x_2 * x_3)$to give us a non feature space. There we can then determine the SVM which will correctly make the non-linear classification.

Kernels are an extension of the above. The extension is that we can compute the dot product of the points in the new feature space without explicitly doing the feature space conversion.
The kernel value K(x,y) between two datapoints x and y is simply the dot product in some feature space. So

K(x,y) =$\Phi(x)^T\Phi(y)$

Since the SVM only requires dot products this means we can run the SVM in the feature space given by Phi just with the kernel values.

Let's look at the polynomial degree two kernel. Suppose we have two data points $x = (x_1, x_2)$and $y = (y_1, y_2)$. Let's transform into new feature space given by Φ that contains the original coordinate and all powers of 2. So

$\Phi(x_1, x_2) \; = \; (1, x_1, x_2, x_1x_2, {x_1}^2, {x_2}^2)$and $\Phi(y_1, y_2) \; = \; (1, y_1, y_2, y_1y_2, {y_1}^2, {y_2}^2)$

$$\Phi^T(x)\Phi \quad (y) \; = \; 1 + x_1y_1 \; + x_2y_2 + x_1x_2y_1y_2 + {x_1}^2{y_1}^2 + {x_2}^2{y_2}^2$$

But we can get the same dot product above with the polynomial degree 2 kernel.

$$K(x,y) = (x^Ty + 1)^2 = (x_1y_1 + x_2y_2 + 1)^2 = 1 + x_1y_1 \; + x_2y_2 + x_1x_2y_1y_2 + {x_1}^2{y_1}^2 + {x_2}^2{y_2}^2$$

And so we see that the kernel gives us the dot product implicitly without explicit feature space conversion.