Exercise 1

1a) Polynomial Kernel: $k(x, x') = (\langle x, x' \rangle + c)^p$

Given: $c = 1$; $p = 2$; $x = (x_1, x_2)$; $x' = (x_1', x_2')$

What is $\phi(x)$?

$$k(x, x') = (\langle x, x' \rangle + 1)^2 = \left(\sum_{i=1}^{2} x_i x_i' + 1\right)^2 = (x_1 x_1' + x_2 x_2' + 1)^2 \quad \Big| \text{ write it out}$$

$$= \left((x_1 x_1')^2 + x_1 x_1' x_2 x_2' + x_1 x_1'\right) + \left(x_2 x_2' x_1 x_1' + (x_2 x_2')^2 + x_2 x_2'\right) + \left(x_1 x_1' + x_2 x_2' + 1\right) \quad \Big| \text{ simplify}$$

$$= (x_1 x_1')^2 + (x_2 x_2') + 2(x_1 x_1' x_2 x_2') + 2(x_1 x_1') + 2(x_2 x_2') + 1$$

$$\phi(x) = \left(x_1^2, x_2^2, \sqrt{2}\, x_1 x_2, \sqrt{2}\, x_1, \sqrt{2}\, x_2, 1\right)$$

The feature space has dimensionality 6.

The feature space of the Gaussian RBF kernel with $\sigma = 1$ has infinite dimensionality. The RBF kernel can be broken down to an infinite sum over polynomial kernels by applying a taylor expansion of e^x. This results in a projection into a vector space with infinite dimensions.

1b) No, we don't need to represent the feature space explicitly for non-linear kernels when using an SVM classifier. We can use the kernel trick as a shortcut around this.

Exercise 2

2a) $\mathbb{X} \in \mathbb{R}^d$; linear kernel : $k(x,x') = \langle x, x' \rangle = \sum_i^d x_i x_i'$

$$\sum_{i,j} c_i c_j k(x_i, x_j) = \sum_{i,j} c_i c_j \langle x_i, x_j \rangle = \sum_{i,j} \langle c_i x_i, c_j x_j \rangle \quad , \text{ for } c_i, c_j \in \mathbb{R}$$

$$= \sum_{i,j} \sum_k c_i x_{i,k} c_j x_{j,k} = \sum_k \left(\sum_i c_i x_{i,k} \right) \left(\sum_j c_j x_{j,k} \right)$$

$$= \sum_k \left(\sum_i c_i x_{i,k} \right)^2 \geq 0$$

-> Condition for positive semi-definitiveness is fulfilled.

2b) $k(x,x') = \langle \phi(x), \phi(x') \rangle$

$$\sum_{i,j} c_i c_j k(x_i, x_j) = \sum_{i,j} c_i c_j \langle \phi(x_i), \phi(x_j) \rangle = \sum_{i,j} \langle c_i \phi(x_i), c_j \phi(x_j) \rangle$$

$$= \sum_{i,j} \sum_k c_i \phi(x_{i,k}) c_j \phi(x_{j,k}) = \sum_k \left(\sum_i c_i \phi(x_{i,k}) \right) \left(\sum_j c_j \phi(x_{j,k}) \right)$$

$$= \sum_k \left(\sum_i c_i \phi(x_{i,k}) \right)^2 \geq 0$$

2c) $k_3(x,x') = k_1(x,x') + k_2(x,x')$

$$\sum_{i,j} c_i c_j k_3(x_i, x_j) = \sum_{i,j} c_i c_j k_1(x_i, x_j) + \sum_{i,j} c_i c_j k_2(x_i, x_j) \geq 0$$

Since k_1 & k_2 are kernels and therefore positive semi-definit (psd), their sum is also psd and therefore a kernel.

$k_4(x,x') = \lambda k_1(x,x')$, $\lambda \in \mathbb{R}^+$

$$\sum_{i,j} c_i c_j k_4(x_i, x_j) = \lambda \sum_{i,j} c_i c_j k_1(x_i, x_j) \geq 0$$

Since k_1 is psd, any multiplication with $\lambda \in \mathbb{R}^+$ will turn out psd.

Exercise 3

3a) $k(x,x') = 6\langle x,x'\rangle^4 + 3 + x^T x' + \exp\left(-\frac{1}{2\sigma^2}\|x-x'\|^2\right)$

$= \lambda_1 k_1\langle x,x'\rangle + \lambda_2 k_2\langle x,x'\rangle + k_3\langle x,x'\rangle + k_4\langle x,x'\rangle$

where: $\lambda_1 = 6$

k_1 = polynomial kernel with $c=0$ and $p=4$

$\lambda_2 = 3$

k_2 = "all-ones" kernel

k_3 = linear kernel

k_4 = Gaussian RBF kernel

→ According to proof in 2c) we can sum all of these up and receive a kernel.

3b)

$k_{base}(s,s') = \begin{cases} 0, & \text{if } s \text{ or } s' \text{ don't start with a 'G'} \\ 1, & \text{if } s \text{ and } s' \text{ start with a 'G' and the values in positions 2 \& 3 of } s \,\&\, s' \text{ differ} \\ 2, & \text{if } s \text{ and } s' \text{ start with a 'G' and there is only one match for values in positions 2 \& 3 of } s \,\&\, s'. \\ 3, & \text{if } s \text{ and } s' \text{ start with a 'G' and the values in positions 2 \& 3 of } s \,\&\, s' \text{ match.} \end{cases}$

The substructures of S and S' are all possible 3-mers of the strings S and S'.

Formal mathematical description of $k_{base}(s,s')$, with python indexing.

$$k_{base}(s,s') = \begin{cases} 0, & \text{if } s[0] \neq G \vee s'[0] \neq G \\ 1, & \text{if } s[0] == s'[0] == G \wedge s[1] \neq s'[1] \wedge s[2] \neq s'[2] \\ 2, & \text{if } s[0] == s'[0] == G \wedge (s[1] == s'[1] \oplus s[2] == s'[2]) \\ 3, & \text{if } s[0] == s'[0] == G \wedge s[1] == s'[1] \wedge s[2] == s'[2] \end{cases}$$

3d)

If we compare sequences of unequal length, where one is much longer than the other, it is hard to tell significance of the resalt, or will be misleoding. It seems to be a similar problem like with Jaccard measures on finite sets.
An attempt to fix this would be to normalize the score.

I do not see a problem if the sequence lengths aren't multiples of 3, it is irrelevant to the task at hand.