EXAM 21.VI.20

Question 1

1). $p(\text{DOWN}) = 1 - p(\text{OUT}) = 0.2$

$p(\text{HIGH}, \text{DOWN}) = p(\text{HIGH}|\text{DOWN})\, p(\text{DOWN}) = 0.18$

$p(\text{LOW}) = 1 - p(\text{HIGH}, \text{DOWN}) - p(\text{HIGH}, \text{OUT}) = 0.72$

$p(\text{LOW}, \text{DOWN}) = p(\text{LOW}|\text{DOWN})\, p(\text{DOWN}) = [1 - p(\text{HIGH}|\text{DOWN})]\, p(\text{DOWN}) = 0.02$

$p(\text{DOWN}|\text{LOW}) = \frac{p(\text{LOW}|\text{DOWN})\, p(\text{DOWN})}{p(\text{LOW})} = 0.0278$

Question 2

1). Given a classification problem with K classes we want to find $y = \tilde{w}^T \tilde{x}$ with y K-class linear discriminant. We can do this iff. the dataset is linearly separable.

2). I think that the correct choice is the line C. In fact SVM aims at maximum margin with the better accuracy and it will always find the optimal solution. In the plot is possible to see that the line C has the maximum margin.

3). Yes, but we have to introduce some slack variables $\xi_n \geq 0 \ \forall n = 1 \dots N$

In this mode the formulation of the problem becomes:

$t_n y(x_n) \geq 1 - \xi_n \ \forall n = 1 \dots N$ and $w^* = \underset{w}{\text{argmin}} \ \frac{1}{2}\|w\|^2 + C \sum_{n=1}^{N} \xi_n$

Question 3

1). In a parametric model we have a fixed number of parameters, in a non-parametric model the number of parameters grows with the amount of data.

2). One example is least squares. We want to find $\tilde{w}^* = \underset{\tilde{w}}{\text{argmin}}\, E(w)$

$E(w) = \frac{1}{2}\{(t - \tilde{X}\tilde{w})^T (t - \tilde{X}\tilde{w})\} \Rightarrow \tilde{w}^* = \tilde{X}^T t$ and $y(x, w) = t^T \tilde{X} + T^T x$

3). One example is K-nn.

- We compute the K nearest neighbors of a new instance $x \notin D$.
- We assign to x the most common label among the majority of neighbors.

$p(C|x, K, D) = \frac{1}{K} \sum_{x_n \in N_K(x_n, D)} \mathbb{I}(C = t_n)$

Question 4

1). We can use a linear regression model. We know that $y(x,w) = \sum_{j=1}^{K} w_j \phi_j(x)$ and the model is still linear in the parameters w. We know that: $t = y(x,w) + \varepsilon$ with ε additive gaussian noise and we know also that $p(\varepsilon|\beta) = N(\varepsilon|0, \beta^{-1}) \longrightarrow p(t|x_1 \dots x_N, w, \beta) = N(t|y(x,w), \beta^{-1})$.

Using the iid hypothesis: $p(\{t_1 \dots t_n\}|x_1 \dots x_N, w, \beta) = \prod_{n=1}^{N} N(t_n|w^T\phi(x_n), \beta^{-1}) =$

$= \sum_{n=1}^{N} \ln N(t_n|w^T\phi(x_n), \beta^{-1}) = -\beta \cdot \frac{1}{2} \sum_{n=1}^{N} (t_n - w^T\phi(x_n))^2 - \frac{N}{2} \ln(2\pi\beta^{-1})$.

We want to find $w^* = \arg\min_w E_D(w)$ with $E_D(w) = \frac{1}{2} \sum_{n=1}^{N} (t_n - w^T\phi(x_n))^2$

2). We can rewrite $E_D(w) = \frac{1}{2}(t - \phi w)^T(t - \phi w)$ and in this mode we obtain $w^* = \phi^+ t$.

Question 5

1). The network takes as input images (matrices of pixels), so the inputs are simply numbers (for example integers). The size of the images can be 24x24 or 32x32 (if in greyscale) or for example 64x64x3 (RGB). The output is instead containing the label which is in the form of one-hot encoded.

2). Convolutional layer that performs the convolution between the image and the kernel. The size of the kernel can be for example 4x4 with padding 0 and stride 2. $\Rightarrow N_{OUT} = \frac{32 - 4 + 0}{2} + 1 = 15$

Then we can use max-pooling. Max-pooling returns the area of a rectangular region. We use a 3x3 kernel with stride 2. $N^*_{OUT} = \frac{15 - 3}{2} + 1 = 7$

At the end we have a fully connected layer so we flatten the network. $N_{FC} = 7 \times \#classes$ with #classes = number of the classes of images in the dataset. The activation function for the hidden layers can be ReLU while for the output layer can be Softmax.

3). The output activation function is softmax because this problem is a typical k-class classification problems. Units saturate with small error.

Question 6

1). The goal of dimensionality reduction is to reduce the size of our dataset in fact a lot of times in our ML problems we have a dataset with very high dimensionality (for example images) but the intrinsic dimensionality is much smaller than the previous one. For example, if we have 12x12 pixels grayscale images we can use PCA to reduce the dimensionality to 3, that is the intrinsic dimensionality of the images, 2 coordinates for the center of the image and the rotation.

2). No, they are two different methods also if the goal is the same. PCA is a linear transformation while autoencoders are used also for non linear functions. Autoencoder is prone to overfitting with high number of parameters A single layered autoencoder with a linear activation function is similar to PCA.

3). We can use Generative Adversarial Networks (GANs). GANs are composed by a geuneretor and a discriminator. Geuneretor produces samples of the distribution $p(x)$ and the discriminator identifies if a sample comes from $p(x)$ or not.