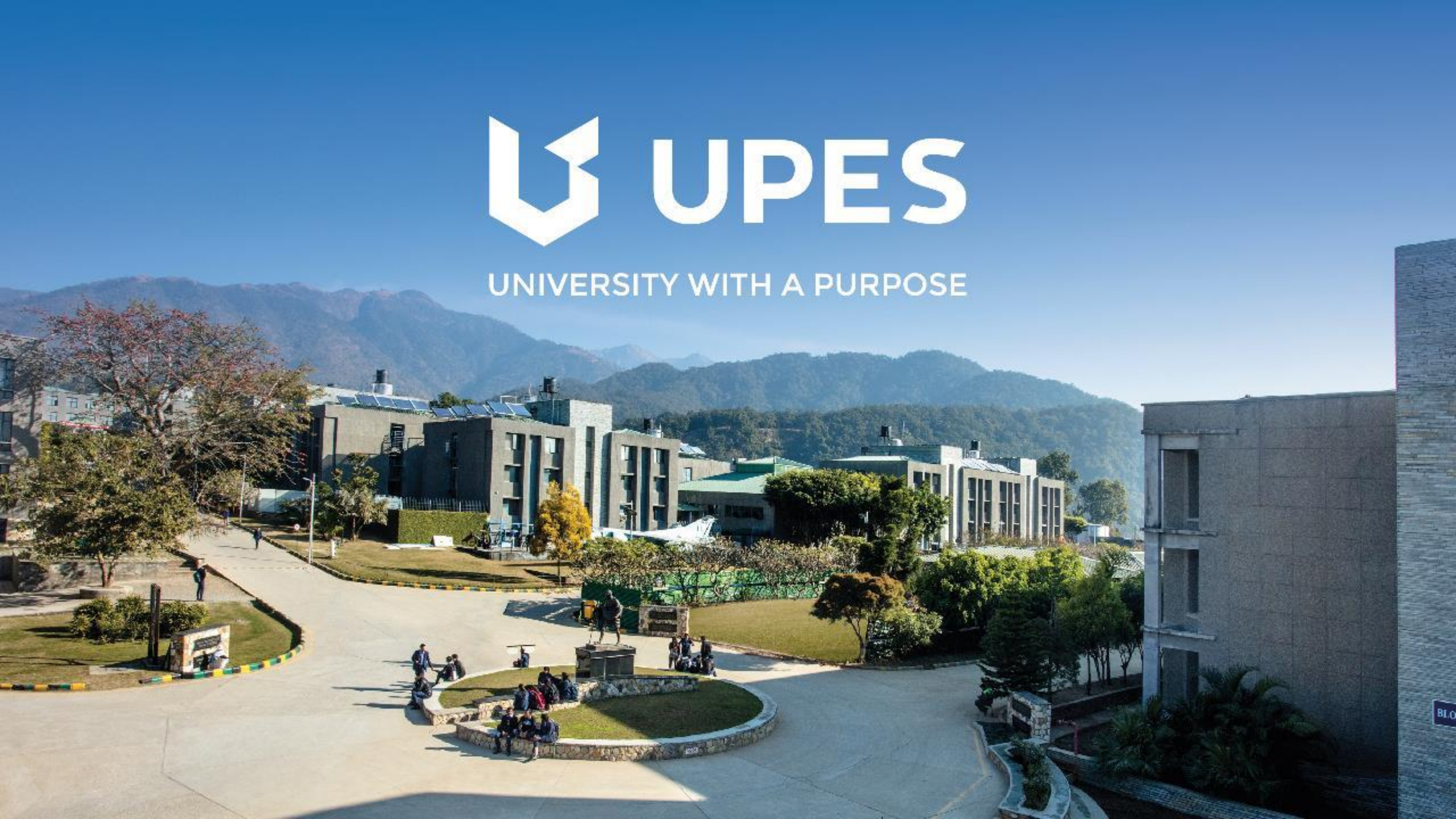
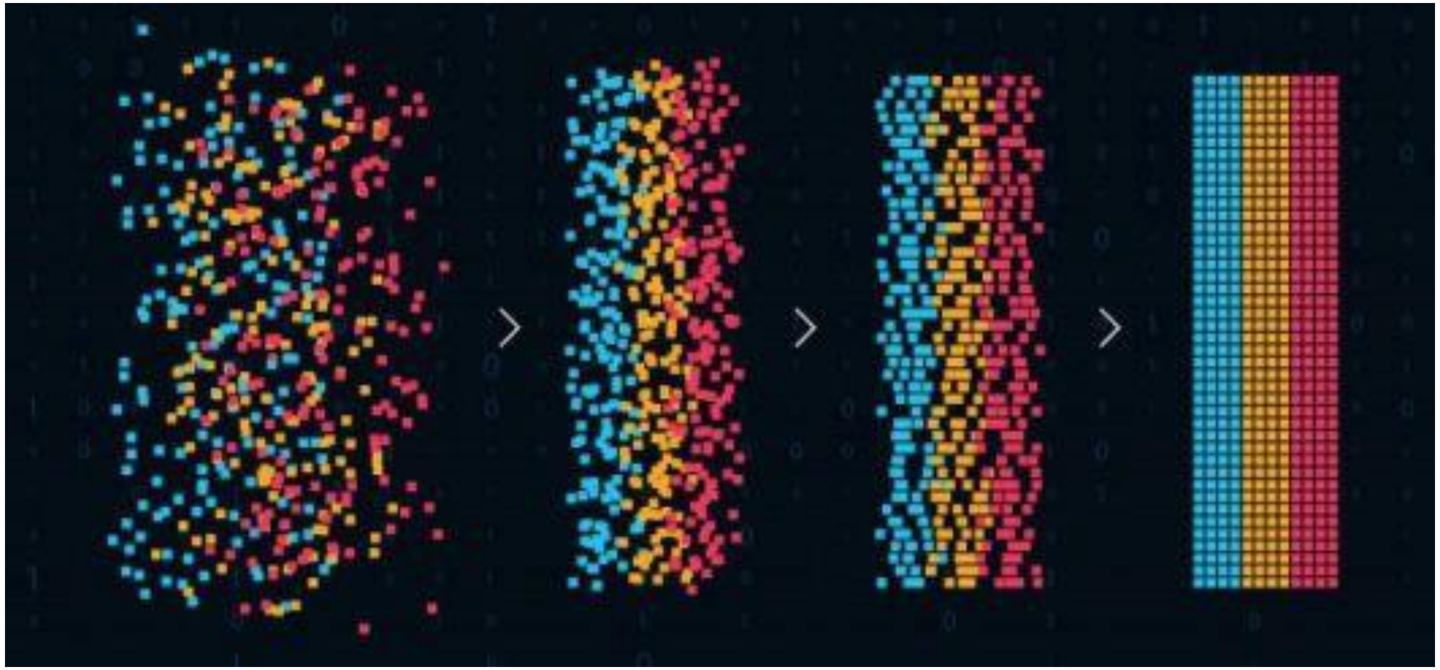

UPES
UNIVERSITY WITH A PURPOSE

Pattern and Anomaly Detection



Source: Edureka

B. Tech., CSE + AI/ML

Dr Gopal Singh Phartiyal

18/11/2021

Recap: Neural Networks: Network Training: Backpropagation

- **Error Backpropagation**
- .

$$\delta_j \equiv \frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k}\frac{\partial a_k}{\partial a_j}$$

$$\delta_j \equiv \frac{\partial E_n}{\partial a_j}$$

$$\delta_j = h'(a_j)\sum_k w_{kj}\delta_k$$

- For batch methods

$$\frac{\partial E}{\partial w_{ji}} = \sum_n \frac{\partial E_n}{\partial w_{ji}}.$$

Recap: Neural Networks: Network Training: Backpropagation

Example: 2-layer network, output layer activation = linear

- .Consider activation function: Tanh
- Its derivative is useful

$$h(a) \equiv \tanh(a)$$

$$\tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}.$$

$$h'(a) = 1 - h(a)^2.$$

- Error function: SSE

$$E_n = \frac{1}{2}\sum_{k=1}^{K}(y_k - t_k)^2$$

Recap: Neural Networks: Network Training: Backpropagation

- Forward pass

$$a_j = \sum_{i=0}^{D} w_{ji}^{(1)} x_i$$

$$z_j = \tanh(a_j)$$

$$y_k = \sum_{j=0}^{M} w_{kj}^{(2)} z_j.$$

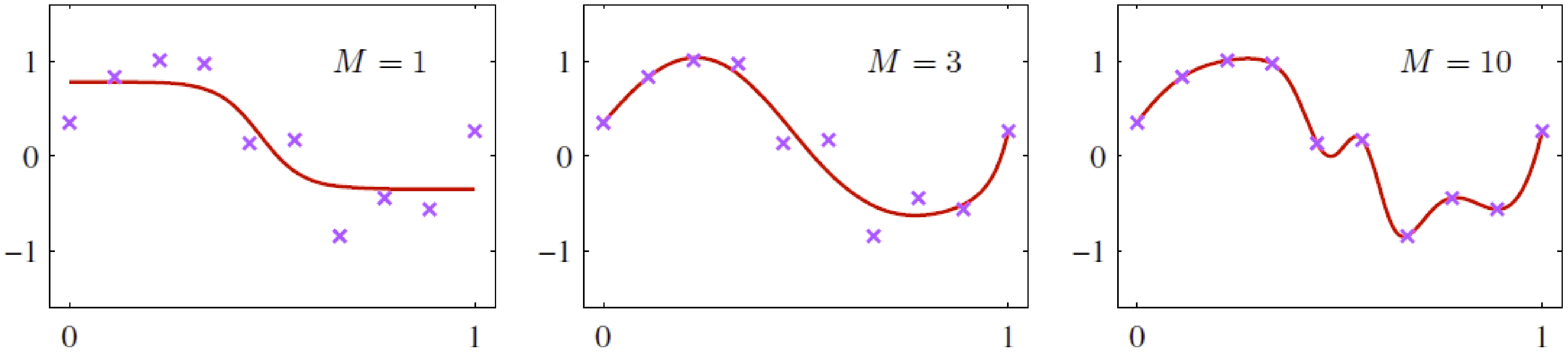
- Backward pass

$$\delta_k = y_k - t_k.$$

$$\delta_j = (1 - z_j^2) \sum_{k=1}^{K} w_{kj} \delta_k.$$

$$\frac{\partial E_n}{\partial w_{ji}^{(1)}} = \delta_j x_i, \qquad \frac{\partial E_n}{\partial w_{kj}^{(2)}} = \delta_k z_j.$$

Neural Networks: Overfitting and Network Regularization



- **L2- Regularization**

$$\widetilde{E}(\mathbf{w}) = E(\mathbf{w}) + \frac{\lambda}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}.$$

- **Consistent Gaussian priors**
- **Early stopping**

Neural Networks: Bayesian Neural Networks: Regression

- **Premise**: Due to highly non-linear dependence of network functions on model parameters (w), it is challenging to build a complete Bayesian treatment unlike the linear regression.
- New concept: Variational inference
- Simple prior: Gaussian (0 mean, variance)

$$p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I}).$$

- Likelihood

$$p(\mathcal{D}|\mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|y(\mathbf{x}_n, \mathbf{w}), \beta^{-1})$$

- Posterior parameter distribution (Non-Gaussian)

$$p(\mathbf{w}|\mathcal{D}, \alpha, \beta) \propto p(\mathbf{w}|\alpha)p(\mathcal{D}|\mathbf{w}, \beta).$$

Neural Networks: Bayesian Neural Networks: Regression

$$p(\mathbf{w}|\mathcal{D}, \alpha, \beta) \propto p(\mathbf{w}|\alpha)p(\mathcal{D}|\mathbf{w}, \beta).$$

- Posterior parameter distribution (Non-Gaussian)
- What to do?
- Approximation (more precisely, Gaussian approximation)
  - Use of Laplace approximation method and
  - Interative numerical optimization

Neural Networks: Bayesian Neural Networks: Classification

- Consider two-class classification first in probabilistic perview
- Consider the following conditional distribution (must look familiar)

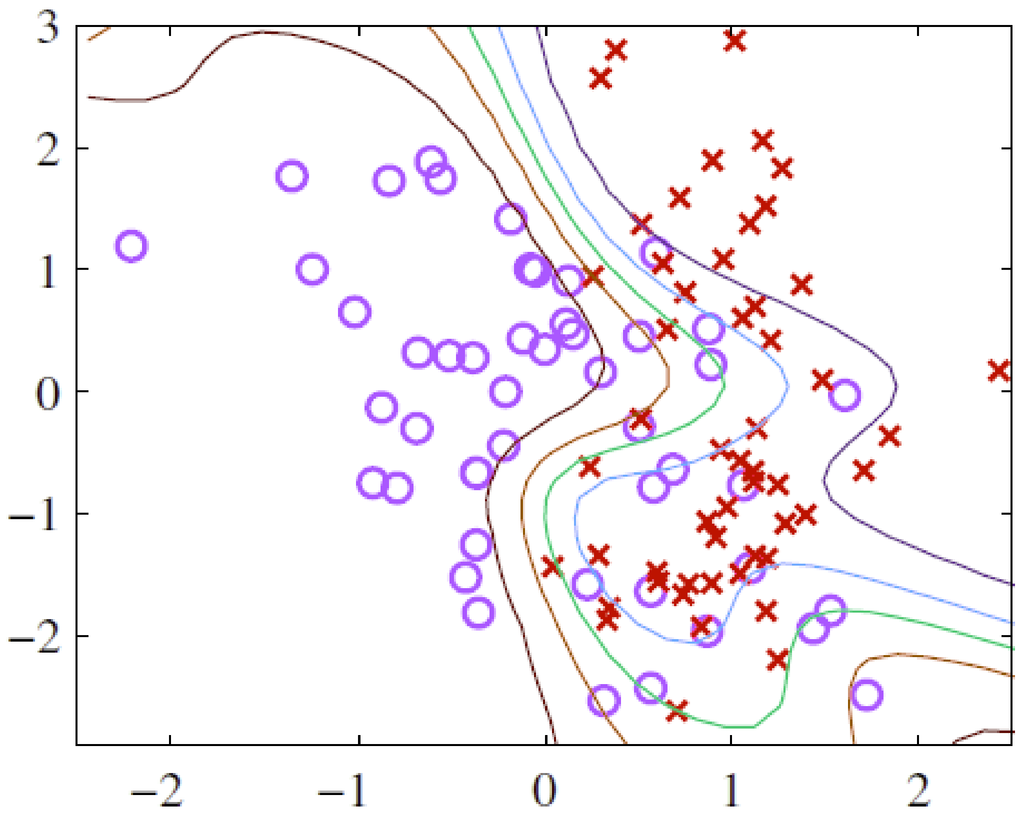
$$\ln p(\mathcal{D}|\mathbf{w}) = \sum_{n} = 1^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

$$E(\mathbf{w}) = -\ln p(\mathcal{D}|\mathbf{w}) + \frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$$

- Now, use gradient descent, and error backpropagation method to solve for w.

Neural Networks: Bayesian Neural Networks: Classification

- Graphical example: two-class
  - Two-dimensional input
  - Two layered NN
  - 8 hidden nodes (tanh)
  - 1 output node (logistic)



Next time: Kernel Models

Thank You

UPES
UNIVERSITY WITH A PURPOSE