1. SVM Loss :

Score Function:

- For an element x_i, & y_i

$$\boxed{\begin{array}{l} s = \text{scores} = f(x_i, W) \\ s_j = \text{score of label } j \text{ (class)} \end{array}}$$

- Loss function: $L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta)$

[max (0, −)
Hinge loss
Hinges at 0, ignore −ve]

Δ = Margin we want s_{y_i} to be greater than all other scores. (Least Margin)

☆ Minimize Loss Function – ML Problem!
To get W — optimisation.

- Regularization: Restrict set of Weights W (Family of functions) because W may not be unique. (2W, 3W, ..., λW)

☆ Full Multiclass SVM Loss, Regularized:

$$L = \underbrace{\frac{1}{N} \sum_i L_i}_{\text{Data Loss}} + \underbrace{\lambda R(W)}_{\text{Regularization Penalty}}$$

λ – hyperparameter (Tune while Training)

Data Loss (Avg. loss L_i of all examples)

Regularization Penalty – Restrict large weights.

$$R(W) = \sum_m \sum_n W_{m,n}^2 \quad \text{(L2 norm)}$$

Softmax Classifier:

☆ Cross Entropy Loss: $L_i = -\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right)$

$$= -f_{y_i} + \log \sum_j e^{f_j}$$

☆ Softmax function: $\boxed{f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}}}$ → Normalize Scores

z – Vector of arbitrary real values.
– Gets squashed to values between 0 & 1 that sum to 1.

• Practical Issue of Numerical Stability:

Due to exponents in softmax, it can be unstable and blow up. So a trick is to do this:

$$\frac{e^{f_{y_i}}}{\sum_j e^{f_j}} = \frac{C e^{f_{y_i}}}{C \sum_j e^{f_j}} = \frac{e^{f_{y_i} + \log C}}{\sum_j e^{f_j + \log C}}$$

Choose $\log C = -\max_j f_j$.

So, for example, [100, 200, 300] becomes
[-200, -100, 0] → easier to calculate.

• Softmax gives probabilities/confidence for each class.
↳ More peaky – Small λ } Regularization
More diffuse – Big λ } Penalty.

☆ SVM vs. Softmax.

SVM more "local", doesnt penalize uncertainty. For eg., if scores are [10, 9, 9], with $\Delta = 1$, the hinge loss is 0 since score of correct class is above margin (1) of other class scores. It would be 0 even for [10, -100, -100].

But, softmax penalizes [10, 9, 9] with higher loss than [10, -100, -100].

☆ Practical Note: Normalize images,

1. Choose a "mean image" and subtract it from every other image. Pixel values are transformed from [0... 255] to [-127... 127] → Centering the mean to 0.

2. Normalize further to have range [-1...1]