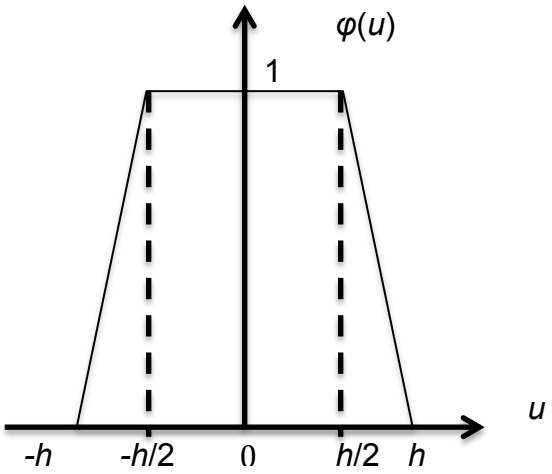
1. The pdf of a $\Gamma(2,1)$ random variable is $f(z) = z\exp(-z),\ z > 0$, and the pmf of a Poisson random variable X is $f_X(x) = \lambda^x e^{-\lambda}/x!,\ \lambda > 0,\ x = 0, 1, \ldots$. Assuming that $X_1, X_2, \ldots, X_n$ is an i.i.d Poisson sample and that λ has a $\Gamma(2,1)$ prior distribution, find the MAP estimate of λ and prove that what you find is actually a value that maximizes the posterior.

2. DHS Chapter 3, Problem 17.

   Hints: Note that $x_i \in \{0, 1\}$ and the problem walks you through constructing MAP estimates. For part (b), the uniform distribution for $p(\theta)$ extends over $0 \leq \theta \leq 1$; also, use Bayes theorem.

3. DHS Chapter 4, Problem 2

4. We perform Parzen Window density estimation, using trapezoidal window functions given (in unnormalized form) in the figure below:



   Choose $h = 1$. Assume that we have the following data

$$\mathcal{D}_{\omega_1} = \{0, 2, 5\}$$
$$\mathcal{D}_{\omega_2} = \{4, 7\}$$

   (a) Sketch or plot the Parzen window estimates of the pdfs $p(x|\omega_1)$ and $p(x|\omega_2)$. Please label pertinent values on both axes.

   (b) Estimate the prior probabilities based on frequency of occurrence of the prototypes in each class.

   (c) Use the estimates you have developed in above to find the decision boundaries and regions for a Bayes minimum-error classifier based on Parzen windows. Only the part of feature space where at least one density is nonzero need to be classified.

5. We perform k-nearest neighbor density estimation, using $k = 2$. Assume that you are given the following training set for a 2-class problem with one feature:

$$\mathcal{D}_{\omega_1} = \{2, 5\}$$
$$\mathcal{D}_{\omega_2} = \{4, 7\}$$

(a) Sketch or plot the k-nearest neighbors estimates of the pdfs $p(x|\omega_1)$. Please label pertinent values on both axes. Also give the density estimates algebraically, for each region in feature space.

(b) Estimate the prior probabilities based on frequency of occurrence of the prototypes in each class.

(c) Use the estimates you have developed in above to find the decision boundaries and regions for a Bayes minimum-error classifier based on k-nearest neighbors.

(d) Derive a classifier based on using KNN as a discriminative technique that estimates $p(\omega_i|x)$ directly using nearest neighbors, and compare it to the classifier you obtained in 5c.

6. For the following classification problem, design a single-layer perceptron that has zero error on training set, by using the multiclass Perceptron update rule.

$$\mathcal{D}_{\omega_1} = \left\{ \mathbf{x}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\}$$
$$\mathcal{D}_{\omega_2} = \left\{ \mathbf{x}_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \right\}$$
$$\mathcal{D}_{\omega_3} = \left\{ \mathbf{x}_3 = \begin{bmatrix} -1 \\ -1 \end{bmatrix} \right\}$$

Use one-hot coding for classes, for example, ω_3 should be represented using the following vector

$$\mathbf{y}_3 = \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix}$$

(a) Start with $\mathbf{W}(0) = \mathbf{0}_{2\times 3}$ and choose $\eta(i) = 0.5$ in $\mathbf{W}(i+1) = \mathbf{W}(i) + \eta(i)\mathbf{x}(i)\mathbf{e}^T$. Do not use the augmented space and assume that the biases are always zero (no update for biases). Show all steps until your algorithm converges. It is alright if you use a computer or calculator to perform the matrix calculations, but you should write down all the steps, and should not write a computer program to yield the final results.

(b) Now redo the previous procedure, but this time deal with each of the columns of $\mathbf{W}$ as one perceptron, i.e. update each column (the weight associated with a

linear discriminant) separately, for example the first iteration becomes:

$$\begin{aligned}
&\mathbf{W}(0) = [\mathbf{w}_1(0)\ \mathbf{w}_2(0)\ \mathbf{w}_3(0)] \\
&\mathbf{w}_1(1) = \mathbf{w}_1(0) + \eta e_1 \mathbf{x}(1) \\
&\mathbf{w}_2(1) = \mathbf{w}_2(0) + \eta e_2 \mathbf{x}(1) \\
&\mathbf{w}_3(1) = \mathbf{w}_3(0) + \eta e_3 \mathbf{x}(1)
\end{aligned}$$

where $e_i = y_{i1} - \text{sign}(\mathbf{w}_i(0)^T \mathbf{x}(1))$ is the difference between the i^{th} element of $\mathbf{y}_1$ (the target vector for $\mathbf{x}_1$) and the output of the i^{th} neuron/linear discriminant $\mathbf{w}_i(1)$.

Perform two epochs only. This is essentially to make you observe that a multicategory Perceptron algorithm is based on multiple binary problems.