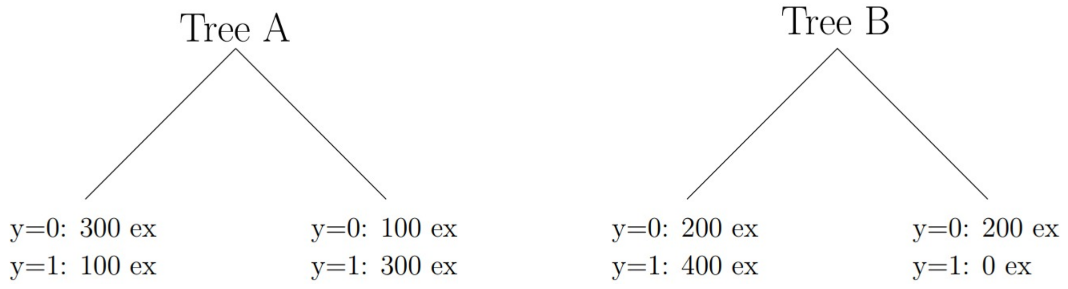
Problem 1 (10 marks)

(Misclassification Rate vs Information Gain) Consider a binary classification data set consisting of 400 data points from class 0 and 400 data points from class 1. Suppose that a decision tree model $\mathcal{A}$ splits these into (300, 100) at the first leaf node and (100, 300) at the second leaf node. (Here, (n, m) denotes that n points are assigned to class 0 and m points are assigned to class 1.) Similarly, suppose that a second decision tree model $\mathcal{B}$ splits them into (200, 400) and (200, 0). (See the figure below.)



(1) Compute the training data **misclassification rate** (i.e., what fraction of training examples will be misclassified) for the two trees: are they equal or not? (2) Evaluate the **information gain** for the two trees and use these to compare the trees. (3) Do you get different answers for (1) and (2)? Does this make sense?

Problem 2 (10 marks)

(Consistent or Not?) An important notion for a classifier is that of *consistency*. A classification algorithm is said to be *consistent* if, whenever it has access to **infinite** amounts of training data, its error rate approaches the optimal error rate (a.k.a. *Bayes optimal*). Consider the noise-free setting (i.e., every training input is labeled correctly). Here, the Bayes optimal error rate is zero. Is the one-nearest-neighbor algorithm consistent in this setting? Briefly justify your answer in 100 words or less.

Problem 3 (10 marks)

(Linear Regression meets Nearest Neighbors) Show that, for the unregularized linear regression model, where the solution $\hat{\boldsymbol{w}} = (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \boldsymbol{y}$, the prediction at a test input $\boldsymbol{x}_*$ can be written as a weighted sum of all the training responses, i.e.,

$$f(\boldsymbol{x}_*) = \sum_{n=1}^{N} w_n y_n$$

Give the expression for the weights w_n's in this case and briefly discuss (<50 words) in what way these weights are different from the weights in a *weighted version* of K nearest neighbors where each w_n typically is the inverse distance of $\boldsymbol{x}_*$ from the training input $\boldsymbol{x}_n$. **Note:** You do not need to give a very detailed expression for w_n (if it makes algebra messy) but you must give a precise meaning as to what w_n depends on and how it is different from the weights in the weighted K nearest neighbors.

Problem 4 (10 marks)

(Feature-Specific ℓ_2 Regularization) The usual ℓ_2 regularized least squares regression objective can be written as $\mathcal{L}(\boldsymbol{w}) = \sum_{n=1}^{N}(y_n - \boldsymbol{w}^\top \boldsymbol{x})^2 + \frac{\lambda}{2}\boldsymbol{w}^\top \boldsymbol{w}$. In this case, the extent of regularization is the same on all the features (and is controlled by λ). Propose an alternative that still uses ℓ_2 regularization on $\boldsymbol{w}$ but the extent of regularization is different for each entry w_d. Write down the objective function and derive the closed form expression for the weight vector $\boldsymbol{w}$.

Problem 5 (20 marks)

(Multi-output Regression with Reduced Number of Parameters) Consider the multi-output regression in which each output $\boldsymbol{y}_n \in \mathbb{R}^M$ in a real-valued vector, rather than a scalar. Assuming a linear model, we can model the outputs as $\mathbf{Y} = \mathbf{XW}$, where $\mathbf{X}$ is the $N \times D$ feature matrix and $\mathbf{Y}$ is $N \times M$ response *matrix* with row n being $\boldsymbol{y}_n^\top$ (note that each column of $\mathbf{Y}$ denotes one of the M responses), and $\mathbf{W}$ is the $D \times M$ **weight matrix**, with its M columns containing the M weight vectors $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_M$. Let's define a squared error loss function $\sum_{n=1}^{N}\sum_{m=1}^{M}(y_{nm} - \boldsymbol{w}_m^\top \boldsymbol{x}_n)^2$, which is just the usual squared error but summed over all the M outputs. Firstly, verify that this can also be written in a more compact notation as $\text{TRACE}[(\mathbf{Y} - \mathbf{XW})^\top(\mathbf{Y} - \mathbf{XW})]$.

Note that this is the same set-up as one of the problems from Practice Set 1 (if you had a chance to look at it :-)). However, here we will assume that the weight matrix $\mathbf{W}$ can be written as a product of two matrices, i.e., $\mathbf{W} = \mathbf{BS}$ where $\mathbf{B}$ is $D \times K$ and $\mathbf{S}$ is $K \times M$ (assume $K < \min\{D, M\}$). Note that there is a benefit of modeling $\mathbf{W}$ this way, since now we need to learn only $K \times (D + M)$ parameters as opposed to $D \times M$ parameters and, if K is small, this can significantly reduce the number of parameters (in fact, reducing the *effective* number of parameters to be learned is another way of regularizing a machine learning model). Note (you can verify) that in this formulation, each $\boldsymbol{w}_m$ can be written as a linear combination of K columns of $\mathbf{B}$.

With the proposed representation of $\mathbf{W}$, the new objective will be $\text{TRACE}[(\mathbf{Y} - \mathbf{XBS})^\top(\mathbf{Y} - \mathbf{XBS})]$ and you need to learn both $\mathbf{B}$ and $\mathbf{S}$. While it is certaintly possible to learn both (one way to do it is using a procedure called "alternating optimization" that we will look at later), for now, let's keep it simple and assume that the matrix $\mathbf{B}$ is known and only $\mathbf{S}$ needs to be estimated. So your problem reduces to

$$\hat{\mathbf{S}} = \arg\min_{\mathbf{S}} \text{TRACE}[(\mathbf{Y} - \mathbf{XBS})^\top(\mathbf{Y} - \mathbf{XBS})]$$

While it is possible (and straighforward) to also include the Frobenious forms of $\mathbf{S}$ in the above objective to regularize $\mathbf{S}$, we will ignore that for brevity/simplicity.

Derive the expression for $\hat{\mathbf{S}}$. Feel free to use results for matrix derivatives (results you will need can be found in Sec. 2.5 of the Matrix Cookbook). Briefly explain how the form of the solution is identifical to the solution of standard multi-output regression (Practice Set 1 problem), but uses a *transformed* version of the inputs $\mathbf{X}$.

Bonus (not for credit, just for practice :-)) Assume $\mathbf{B}$ is also unknown and find its estimate as well. I already mentioned that one way to do so is by using an "alternating" procedure that opimizes the above objective w.r.t. $\mathbf{S}$ with $\mathbf{B}$ fixed, and then w.r.t. $\mathbf{B}$ with $\mathbf{S}$ fixed (and repeating until convergence). Does the optimal $\mathbf{B}$ has closed form solution, when $\mathbf{S}$ is fixed?