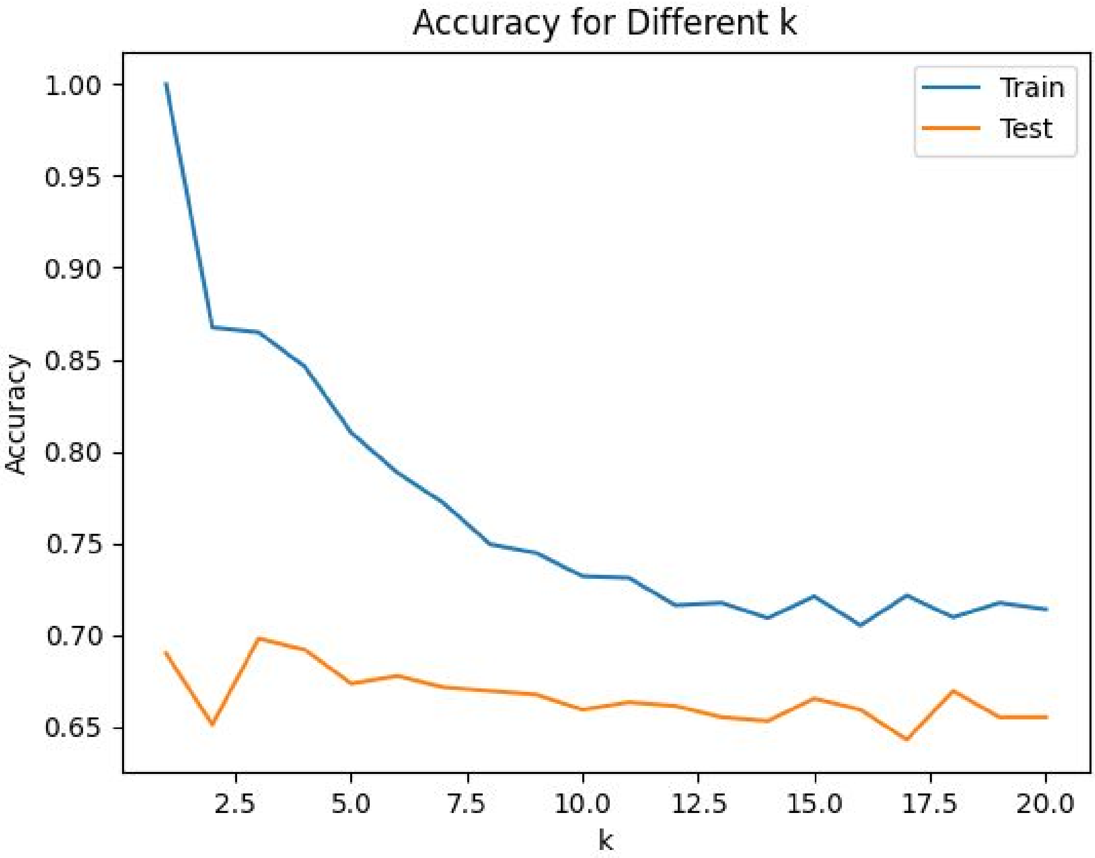
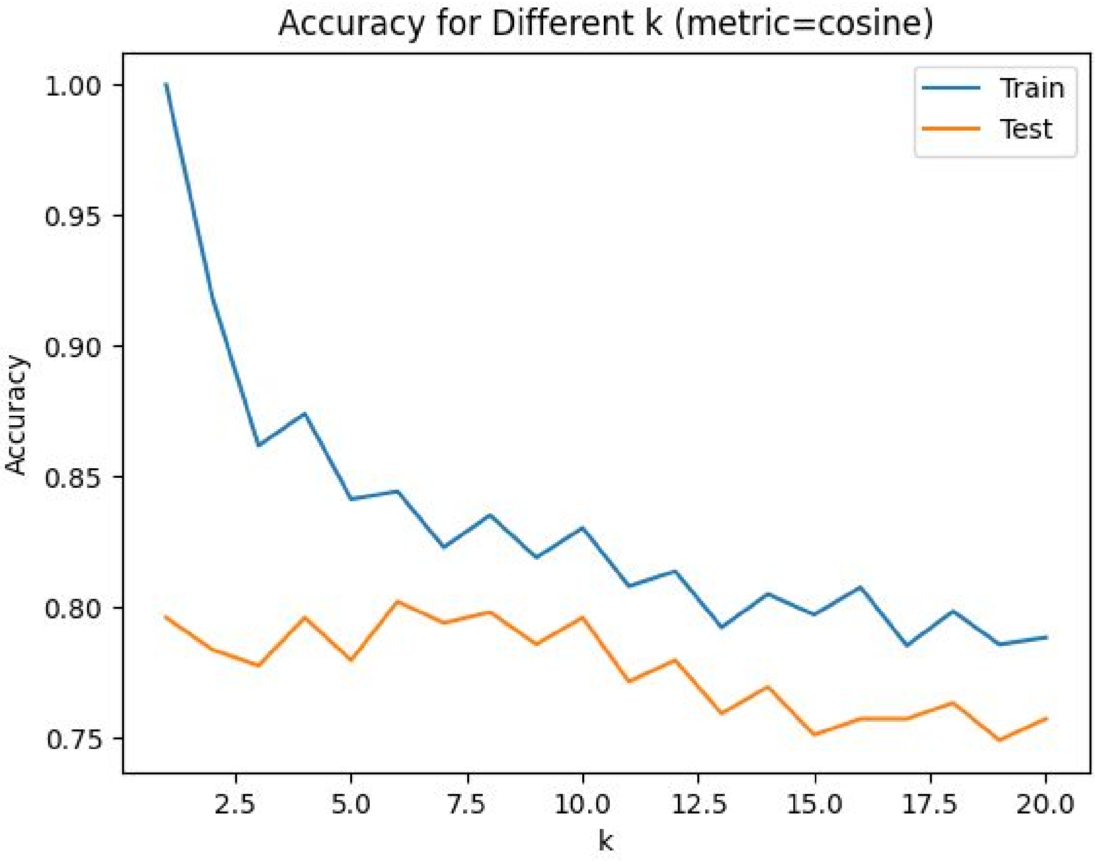
Classification with Nearest Neighbours



Based on the plot, the model with the best validation accuracy was with k=4. When using this value of k with the test data, the model had an accuracy of 0.6857142857142857. We can see that the training accuracy decreases as k increases, which is to be expected because for k=1, we are looking at the point itself. However as k increases, the model becomes too general and fails to take into account the specifics of each data point. We can also see that after the test accuracy peaks, it begins to decrease for the same reason. The reason the test accuracy increases before this happens is because for low values of k, the model overfits to the training data and is not yet generalizable enough to the test data.



Regularized Linear Regression

(a) We have that $\frac{\partial}{\partial w_j}(\frac{1}{2N}\sum_{i=1}^{N}(y^{(i)}-t^{(i)})^2+\frac{1}{2}\sum_{j=1}^{D}\beta_j w_j^2)=\frac{1}{2N}\sum_{i=1}^{N}\frac{\partial}{\partial w_j}(y^{(i)}-t^{(i)})^2+\frac{1}{2}\frac{\partial}{\partial w_j}\sum_{j=1}^{D}(\beta_j w_j^2)=$

$\frac{1}{N}\sum_{i=1}^{N}((y^{(i)}-t^{(i)})\frac{\partial}{\partial w_j}(y^{(i)}-t^{(i)}))+\frac{1}{2}\frac{\partial}{\partial w_j}\sum_{j=1}^{D}(\beta_j w_j^2)=$ *by the chain rule*

$\frac{1}{N}\sum_{i=1}^{N}((y^{(i)}-t^{(i)})\frac{\partial}{\partial w_j}(\sum_{j=1}^{D}(w_j x_j^{(i)}+b)-t^{(i)}))+\frac{1}{2}\frac{\partial}{\partial w_j}\sum_{j=1}^{D}(\beta_j w_j^2)=$ *expanding* $y^{(i)}$

$\frac{1}{N}\sum_{i=1}^{N}((y^{(i)}-t^{(i)})(\frac{\partial}{\partial w_j}(w_j x_j^{(i)}+b)))+\frac{1}{2}\frac{\partial}{\partial w_j}\sum_{j=1}^{D}(\beta_j w_j^2)=$ *We can remove the summation because we only care about a particular* w_j

$\frac{1}{N}\sum_{i=1}^{N}((y^{(i)}-t^{(i)})(x_j^{(i)}))+\frac{1}{2}\frac{\partial}{\partial w_j}\sum_{j=1}^{D}(\beta_j w_j^2)=$

$\frac{1}{N}\sum_{i=1}^{N}((y^{(i)}-t^{(i)})(x_j^{(i)}))+\frac{1}{2}\frac{\partial}{\partial w_j}(\beta_j w_j^2)=$

$\frac{1}{N}\sum_{i=1}^{N}((y^{(i)}-t^{(i)})(x_j^{(i)}))+\frac{1}{2}\beta_j\frac{\partial}{\partial w_j}w_j^2=$

$\frac{1}{N}\sum_{i=1}^{N}((y^{(i)} - t^{(i)})(x_j^{(i)})) + \beta_j w_j$

Now for b, $\frac{\partial}{\partial b}(\frac{1}{2N}\sum_{i=1}^{N}(y^{(i)} - t^{(i)})^2 + \frac{1}{2}\sum_{j=1}^{D}\beta_j w_j^2) = \frac{\partial}{\partial b}(\frac{1}{2N}\sum_{i=1}^{N}(\sum_{j=1}^{D}(w_j x_j^{(i)} + b) - t^{(i)})^2) =$ *expanding* $y^{(i)}$

$(\frac{1}{2N}\sum_{i=1}^{N}\frac{\partial}{\partial b}(\sum_{j=1}^{D}(w_j x_j^{(i)} + b) - t^{(i)})^2) =$

$(\frac{1}{N}\sum_{i=1}^{N}((y^{(i)} - t^{(i)})\frac{\partial}{\partial b}(\sum_{j=1}^{D}(w_j x_j^{(i)} + b) - t^{(i)}))) =$ *by the chain rule*

$\frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - t^{(i)})\frac{\partial}{\partial b}\sum_{j=1}^{D} b =$

$\frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - t^{(i)})\frac{\partial}{\partial b}Db =$

$\frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - t^{(i)})D$

Thus, our update rules are $w_j \leftarrow w_j - \frac{\alpha}{N}\sum_{i=1}^{N}((y^{(i)} - t^{(i)})(x_j^{(i)})) - \beta_j w_j$ and

$b \leftarrow b - \frac{\alpha}{N}\sum_{i=1}^{N}(y^{(i)} - t^{(i)})D$

Notice that the former update rules can be rewritten as $w_j \leftarrow (1 - \frac{\alpha}{N}\beta_j)w_j - \frac{\alpha}{N}\sum_{i=1}^{N}((y^{(i)} - t^{(i)})(x_j^{(i)}))$, showing why it is called weight decay, as w_j is being decayed towards 0 whenever the rule is applied, as we multiply w_j by a number less than 1 in the first half of the expression. The rate of decay depends on $\frac{\alpha}{N}\beta_j$

(b) Using our solution to (a), we want to solve $\frac{1}{N}\sum_{i=1}^{N}((y^{(i)} - t^{(i)})(x_j^{(i)})) + \beta_j w_j = \sum_{j'=1}^{D} A_{jj'}w_{j'} - c_j = 0$ for $A_{jj'}$ and c_j. We have $\sum_{i=1}^{N}((y^{(i)} - t^{(i)})(x_j^{(i)})) = -\beta_j w_j N \rightarrow$

$\sum_{i=1}^{N}((\sum_{j'=1}^{D} w_{j'}x_{j'}^{(i)} - t^{(i)})(x_j^{(i)})) = -\beta_j w_j N \rightarrow$ *expanding* $y^{(i)}$

$\sum_{i=1}^{N}(\sum_{j'=1}^{D} x_j^{(i)}w_{j'}x_{j'}^{(i)} - t^{(i)}x_j^{(i)}) = -\beta_j w_j N \rightarrow$ *distributing* $x_j^{(i)}$ (Notice that the j subscript here is different from the j' subscript on the inner summation)

$\sum_{i=1}^{N}\sum_{j'=1}^{D} x_j^{(i)}w_{j'}x_{j'}^{(i)} - \sum_{i=1}^{N} t^{(i)}x_j^{(i)} = -\beta_j w_j N \rightarrow$ *distributing the summation*

$\sum_{j'=1}^{D}\sum_{i=1}^{N} x_j^{(i)}w_{j'}x_{j'}^{(i)} - \sum_{i=1}^{N} t^{(i)}x_j^{(i)} = -\beta_j w_j N$

Thus, we have $A_{jj'} = \sum_{i=1}^{N} x_j^{(i)}x_{j'}^{(i)}$ and $c_j = \sum_{i=1}^{N} t^{(i)}x_j^{(i)} - \beta_j w_j N$

(c) By getting rid of the j subscripts, we are now looking at a vector equation: $A = \sum_{i=1}^{N} \vec{x}^{(i)} x_{j'}^{(i)}$ and $\vec{c} = \sum_{i=1}^{N} t^{(i)} \vec{x}^{(i)} - \beta \vec{w} N$. We can write then write $\vec{w} = (\sum_{i=1}^{N} \vec{x}^{(i)} x_{j'}^{(i)} \sum_{j'=1}^{D} w_{j'} - \sum_{i=1}^{N} t^{(i)} \vec{x}^{(i)}) / - \beta_j N$

Loss Functions

$\vec{y} = \vec{w}^T \vec{x} + b$

$\frac{\partial J}{\partial \vec{y}} = \frac{\partial}{\partial \vec{y}} (\frac{1}{N} \sum_{i=1}^{N} (1 - cos(y^{(i)} - t)) = \frac{1}{N} \frac{\partial}{\partial \vec{y}} (1 - cos(y^{(i)} - t)) =$ *We can remove the summation because we only care about a particular* $y^{(i)}$ *for each component of* $\vec{y}$

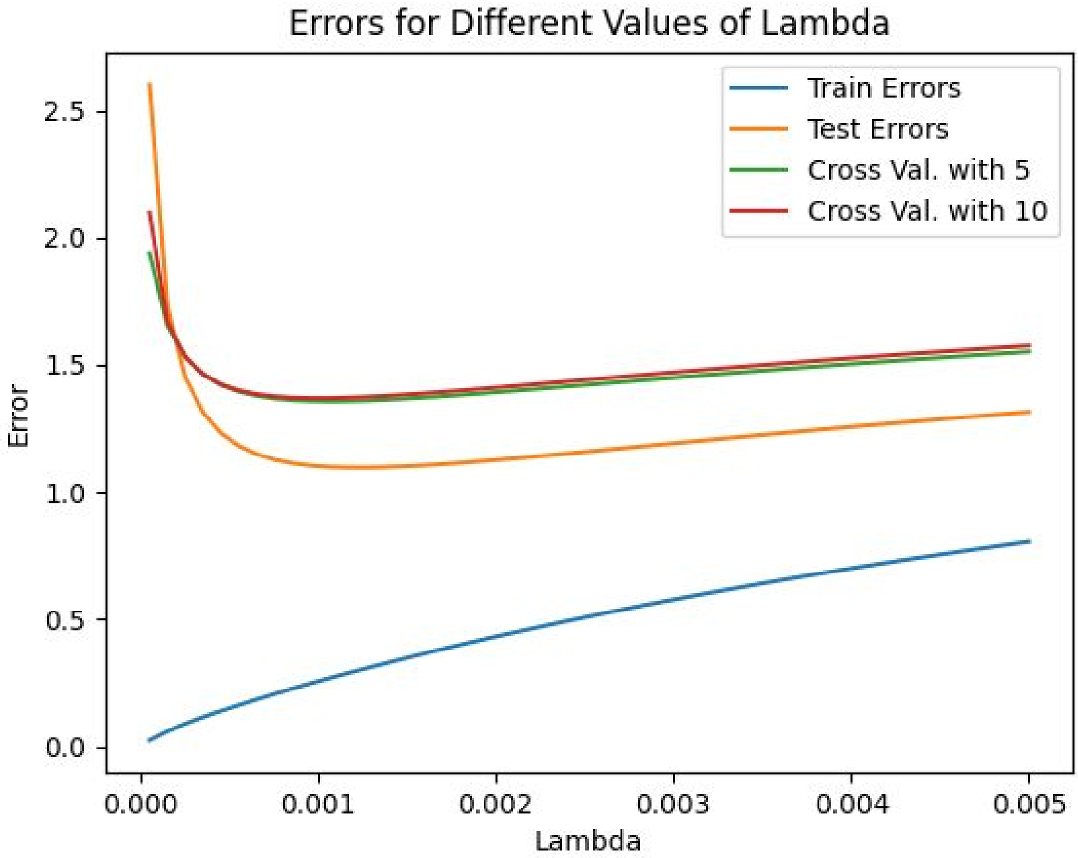
$- \frac{1}{N} \frac{\partial}{\partial \vec{y}} (cos(y^{(i)} - t)) = \frac{1}{N} sin(y^{(i)} - t) \frac{\partial}{\partial \vec{y}} (y^{(i)} - t) =$ *by the chain rule*

$\frac{1}{N} sin(\vec{y} - t)$

$\frac{\partial J}{\partial \vec{w}} = \frac{\partial J}{\partial \vec{y}} \frac{\partial \vec{y}}{\partial \vec{w}} = \frac{1}{N} sin(\vec{y} - t) jac_{\vec{w}}(\vec{y})$, where $jac_{\vec{w}}(\vec{y})$ denotes the Jacobian matrix of $\vec{y}$ with partial derivatives with respect to the components of $\vec{w}$: Row 1: $[\frac{\partial y_1}{\partial w_1}, \frac{\partial y_1}{\partial w_2}, ..., \frac{\partial y_1}{\partial w_n}]$ and Column 1 (as a row vector): $[\frac{\partial y_1}{\partial w_1}, \frac{\partial y_2}{\partial w_1}, ..., \frac{\partial y_n}{\partial w_1}]$

$\frac{\partial J}{\partial b} = \frac{\partial J}{\partial \vec{y}} \frac{\partial \vec{y}}{\partial b} = \frac{1}{N} sin(\vec{y} - t) \vec{1}$, where $\vec{1}$ is a vector composed entirely of 1’s

Cross Validation



Looking at the plot, the optimal value of lambda seems to be around 0.0005. With non-trained data, the errors all seem to reach a minimum at this value before gradually increasing.