HW2 Writeup

Shiven Taneja 1005871013

October 2021

1 Expected Loss and Bayes Optimality

1.a

Expected loss for the policy that keeps every email: $E(\mathcal{L}(y,t)) = \sum_y \mathcal{L}(y = Keep, t) * Pr(t = Spam) = 0 * (0.9) + 1 * (0.1) = 0.1$
Expected loss for the policy that removes every email: $E(\mathcal{L}(y,t)) = \sum_y \mathcal{L}(y = Remove, t) * Pr(t = Spam) = 100 * (0.9) + 0 * (0.1) = 90$

1.b

In order to determine the Bayes optimal decision rule, we have to find the values that minimize the expected loss given x. That means that $y_* = argmin(p(t|x)) = argmin(E(\mathcal{L}(y,t)))$.
This then becomes $y_* = argmin(E(\mathcal{L}(y,t))) = \sum_t \mathcal{L}(y,t) * p(t|x)$
We can then classify the optimal decision rule by whether the email is spam or nonspam as shown:
$y_*^{Spam} = \sum_t \mathcal{L}(y, t = Spam) * p(t = Spam|x) = \mathcal{L}(y = Keep, t = Spam) * p(t = Spam|x) + \mathcal{L}(y = Remove, t = Spam) * p(t = Spam|x)$
and for Nonspam it is:
$y_*^{Nonspam} = \sum_t \mathcal{L}(y, t = Nonspam) * p(t = Nonspam|x) = \mathcal{L}(y = Keep, t = Nonspam) * p(t = Nonspam|x) + \mathcal{L}(y = Remove, t = Nonspam) * p(t = Nonspam|x)$
We can then subsitute the values so that y_*^{Spam} becomes $\mathcal{L}(y = Keep, t = Spam) * p(t = Spam|x) + \mathcal{L}(y = Remove, t = Spam) * p(t = Spam|x) = 1 * p(t = Spam|x) + 0 * p(t = Spam|x) = p(t = Spam|x)$
and for $y_*^{Nonspam}$
$\mathcal{L}(y = Keep, t = Nonspam) * p(t = Nonspam|x) + \mathcal{L}(y = Remove, t = Nonspam) * p(t = Nonspam|x) = 0 * p(t = Nonspam|x) + 100 * p(t = Nonspam|x) = 100 * p(t = Nonspam|x)$
In order to bayes optimal decision we need to find the minimum of these two functions such that if $y_*^{Spam} < y_*^{Nonspam}$ then our optimal decision becomes y_*^{Spam}

1.c

In order to determine the Bayes optimal decision rule, we have to minimize the probability of error by maximizing $p(t|x)$ as this would ensure the probability of choosing the wrong type of email (spam vs nonspam) is minimized.
We first set $y_* = p(t|x)$ for both $t = spam$ and $t = nonspam$ s.t. $y_*^{Spam} = p(t = Spam|x)$, $y_*^{Nonspam} = p(t = Nonspam|x)$.

In order to find the Bayes optimal decision rule, we find the t value which maximizes y_* s.t. if $y_*^{Spam} > y_*^{Nonspam}$ then y_*^{Spam} is the optimal decision else it is $y_*^{Nonspam}$.
We can further simplify y_* using Bayes rule for conditional probability as shown: $y_* = p(t|x) = \frac{p(x|t)*p(t)}{p(x)}$.
We can now simplify this further for y_*^{Spam} and $y_*^{Nonspam}$ which becomes: $y_*^{Spam} = p(t = Spam|x) = \frac{p(x|t=Spam)*p(t=Spam)}{p(x)} = \frac{p(x|t=Spam)*p(t=Spam)}{p(x|t=Spam)*p(t=Spam)+p(x|t=Nonspam)*p(t=Nonspam)} = \frac{p(x|t)*0.1}{p(x|t=Spam)*p(t=Spam)+p(x|t=Nonspam)*p(t=Nonspam)}$ and $y_*^{Nonspam} = p(t = Nonspam|x) = \frac{p(x|t=Nonspam)*p(t=Nonspam)}{p(x|t=Spam)*p(t=Spam)+p(x|t=Nonspam)*p(t=Nonspam)} = \frac{p(x|t=Nonspam)*0.9}{p(x|t=Spam)*p(t=Spam)+p(x|t=Nonspam)*p(t=Nonspam)}$

We can now find the optimal decision rule for each value of x. For $x_1 = 0$ and $x_2 = 0$ the optimal decision rule is given by:
$y_*^{Spam} = \frac{0.4*0.1}{0.4*0.1+0.998*0.9} = \frac{0.04}{0.04+0.8982} = 0.043$
$y_*^{Nonspam} = \frac{0.998*0.9}{0.4*0.1+0.998*0.9} = \frac{0.8982}{0.04+0.8982} = 0.957$
Thus since $y_*^{Nonspam} > y_*^{Spam} = 0.957 > 0.043$ we know that the Bayes optimal decision for $x_1 = 0$ and $x_2 = 0$ is that the email is non-spam

For $x_1 = 0$ and $x_2 = 1$ the optimal decision rule is given by:
$y_*^{Spam} = \frac{0.3*0.1}{0.3*0.1+0.9*0.001} = \frac{0.03}{0.03+0.0009} = 0.971$
$y_*^{Nonspam} = \frac{0.9*0.001}{0.9*0.001+0.3*0.1} = \frac{0.0009}{0.0009+0.03} = 0.0291$
Thus since $y_*^{Nonspam} < y_*^{Spam} = 0.0291 < 0.971$ we know that the Bayes optimal decision for $x_1 = 0$ and $x_2 = 1$ is that the email is spam

For $x_1 = 1$ and $x_2 = 0$ the optimal decision rule is given by:
$y_*^{Spam} = \frac{0.2*0.1}{0.2*0.1+0.9*0.001} = \frac{0.02}{0.02+0.0009} = 0.957$
$y_*^{Nonspam} = \frac{0.9*0.001}{0.9*0.001+0.2*0.1} = \frac{0.0009}{0.0009+0.02} = 0.043$
Thus since $y_*^{Nonspam} < y_*^{Spam} = 0.043 < 0.957$ we know that the Bayes optimal decision for $x_1 = 1$ and $x_2 = 0$ is that the email is spam

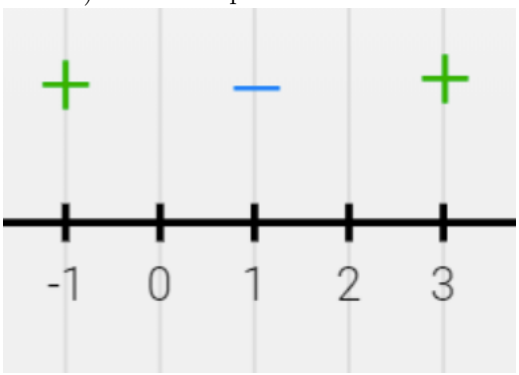
For $x_1 = 1$ and $x_2 = 1$ the optimal decision rule is given by:
$y_*^{Spam} = \frac{0.1*0.1}{0.1*0.1+0.9*0} = \frac{0.01}{0.01+0} = 1$
$y_*^{Nonspam} = \frac{0.9*0}{0.9*0+0.1*0.1} = \frac{0}{0+0.02} = 0$
Thus since $y_*^{Nonspam} < y_*^{Spam} = 0 < 1$ we know that the Bayes optimal decision for $x_1 = 1$ and $x_2 = 1$ is that the email is spam

1.d TO BE COMPLETED

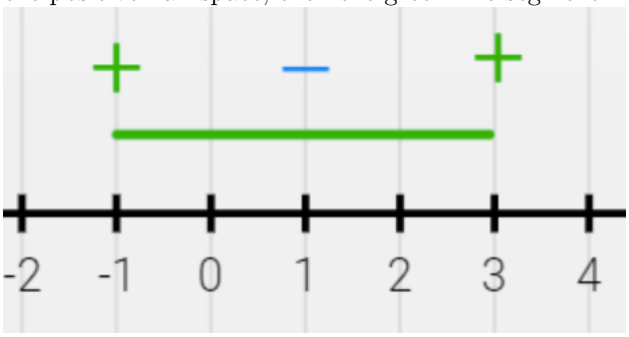
2 Feature Maps

2.a

Showing that the 1-D Data-set for Binary Classification is not linearly separable (proof by contradiction). We can represent the 1-D Data-set by the number line below.



In order for this data-set to be linearly separable, we must be able to create a decision boundary that would separate the space into two half spaces (positive and negative). We further know that if two points lie in a half space, the line segment connecting them must also lie in the same half space. Suppose that there was some feasible weights (hypothesis). If the positive examples are in the positive half-space, then the green-line segment must be as well.



We know that the blue negative point must lie within the negative half-space. Since the green line segment passes through the point (1) which lies within the negative half-space then the data set cannot be linearly separable as the green (positive) line cannot intersect with the negative half-space.

2.b

Applying the feature map to the data-set we get the following data-set:

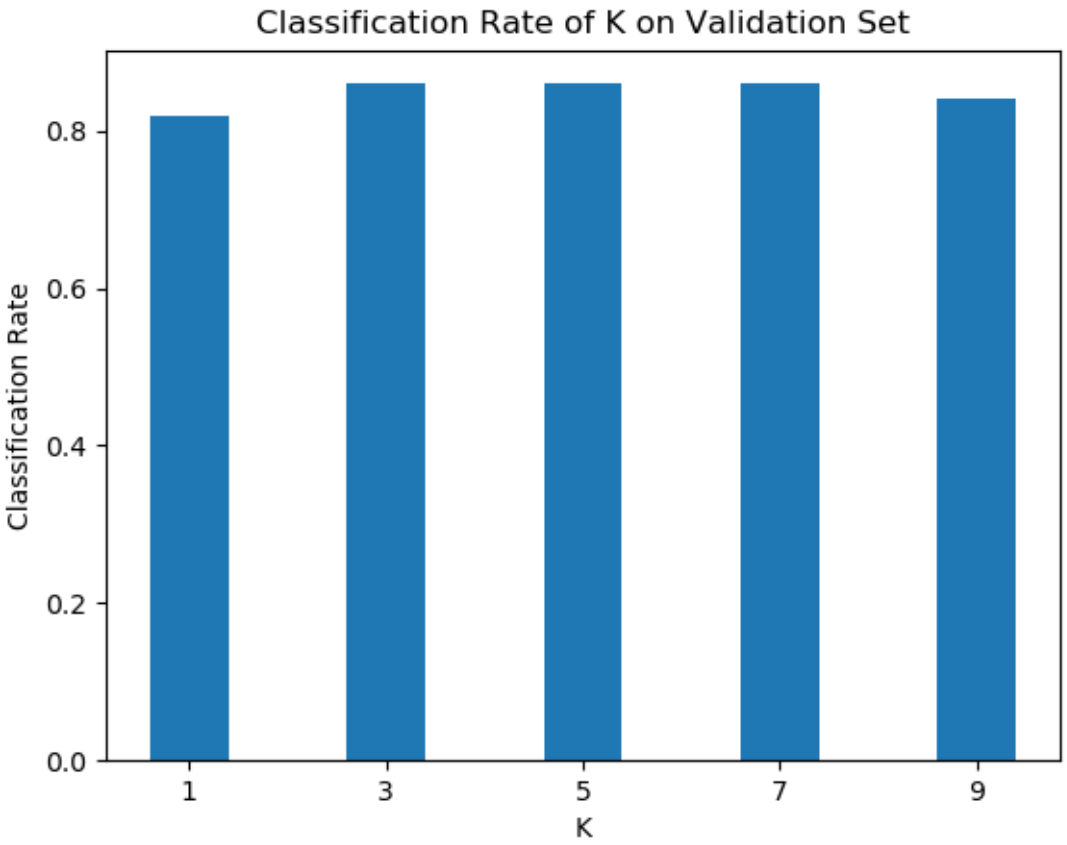
x	$\psi_1(x)$	$\psi_2(x)$	t
-1	-1	1	1
1	1	1	0
3	3	9	1

We can now write down the constraints on ω_1 and ω_2 such that:

$z = \psi_1(x) * \omega_1 + \psi_2(x) * \omega_2$
For $x_1 = -1$ we get that: $\psi_1(x) = -1$ and $\psi_2(x) = 1$ which gives us:
$z = -\omega_1 + \omega_2 \geq 0$
For $x_1 = 1$ we get that: $\psi_1(x) = 1$ and $\psi_2(x) = 1$ which gives us:
$z = \omega_1 + \omega_2 < 0$
For $x_1 = 3$ we get that: $\psi_1(x) = 3$ and $\psi_2(x) = 9$ which gives us:
$z = 3 * \omega_1 + 9 * \omega_2 \geq 0$
Thus solving for ω_1 and ω_2 we get that:
$\omega_1 = -2$
$\omega_2 = 1$
which ensures all conditions are met and thus makes the data-set linearly separable

3 kNN vs Logistic Regression

3.1.a



3.1.b

The classification rates were quite similar for all values of K, with k=1 yielding a classification rate of 0.82 while k=3, 5, 7 yielded a classification rate of 0.86. For k=9 the classification rate was 0.84. Thus 3 of the k values had the same classification rate which was also the highest. Thus I am picking $k^* = 5$ but the ideal classification rate could be either k=3, 5, or 7 as they all have the highest classification rates.
The classification rate for $k^* + 2 = 7$ is also 0.86 and for $k^* - 2 = 3$ is also 0.86.
We can now find the classification rate for the test data. For k = 3 on the test data, the classification rate was 0.92. For k = 5, the classification rate was 0.94 and for k = 7 the classification rate was also 0.94. All of the classification rates checked for the test data set are higher than that of the validation data sets even for the same values of k.

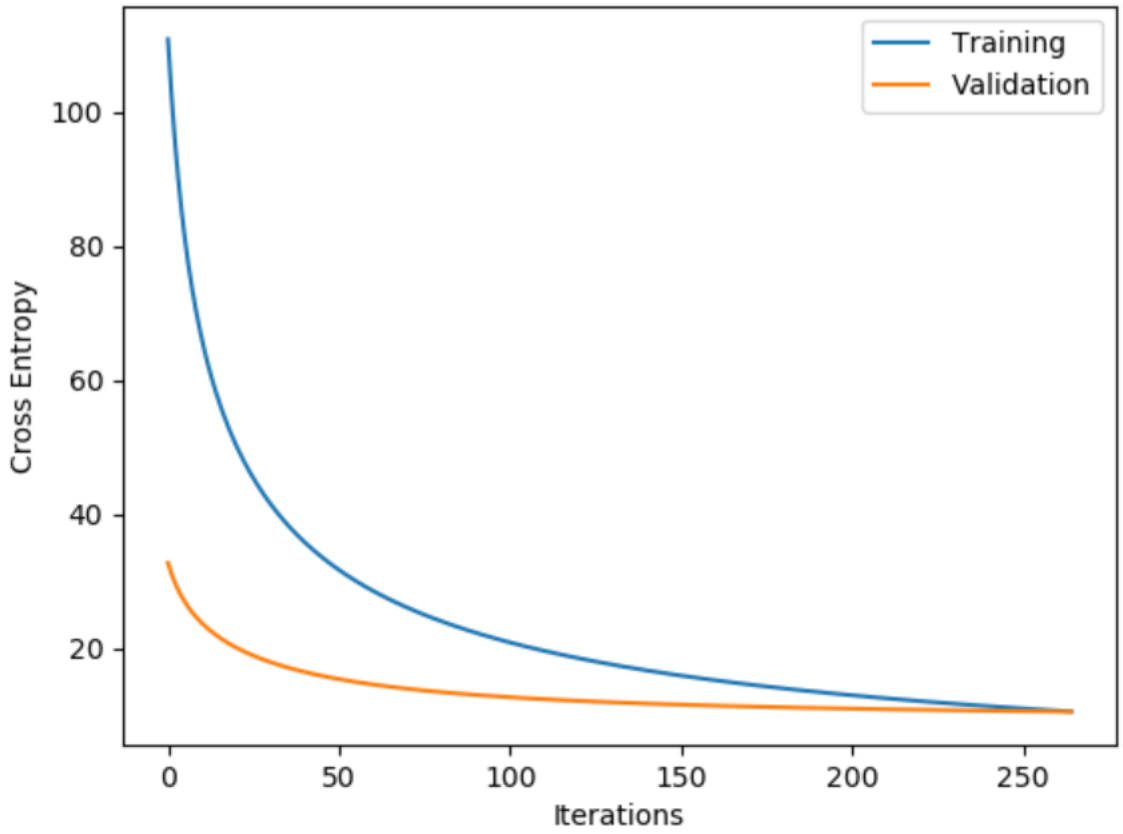
3.2.b

For mnist_train, the hyperparameters chosen was a learning rate of 0.1 and the number of iterations are 265. Using these hyperparameters, the convergence of the two lines was minimized.

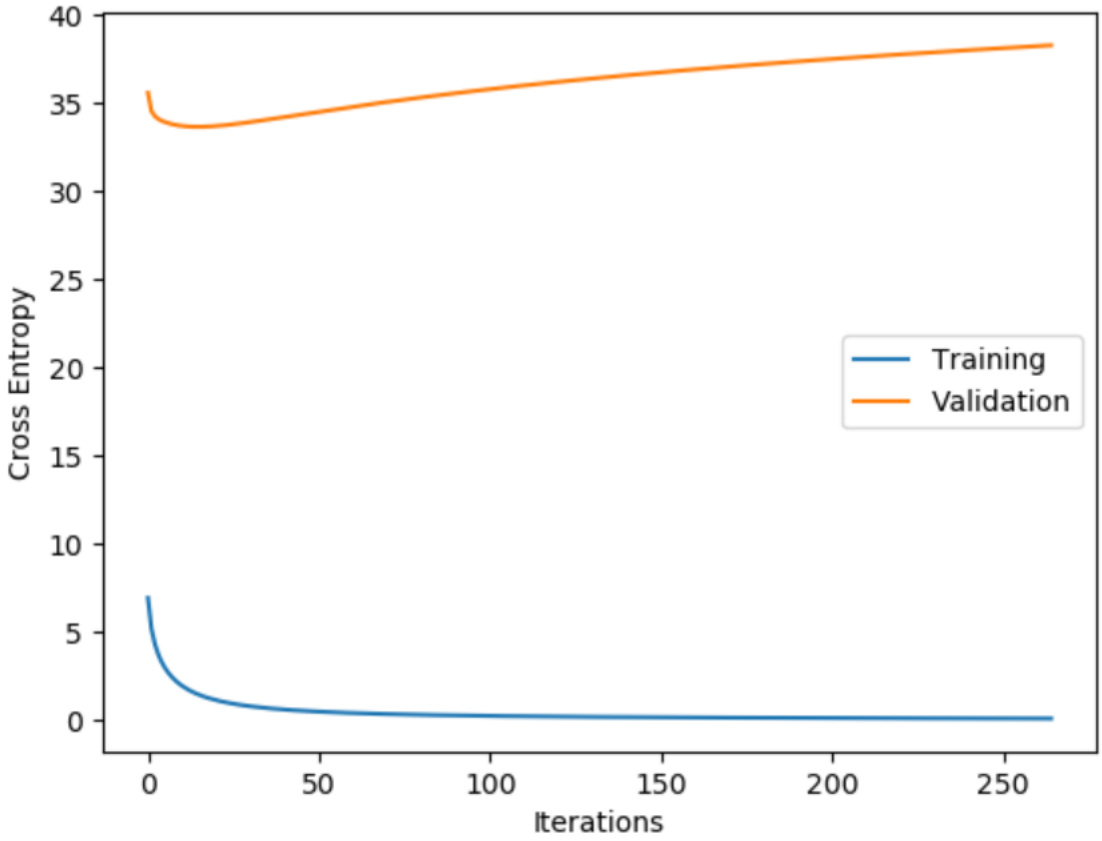
Using these hyperparameters, the final cross entropy on the training set was 10.65 with a classification error of 0. For the validation set, the final cross entropy was 10.499 with a classification error of 0.12. For the test set, the final cross entropy was 10.16 with a classification error of 0.08. The learning rate minimizes the classification error rates.

3.2.c

For mnist_train:



For mnist_train_small:



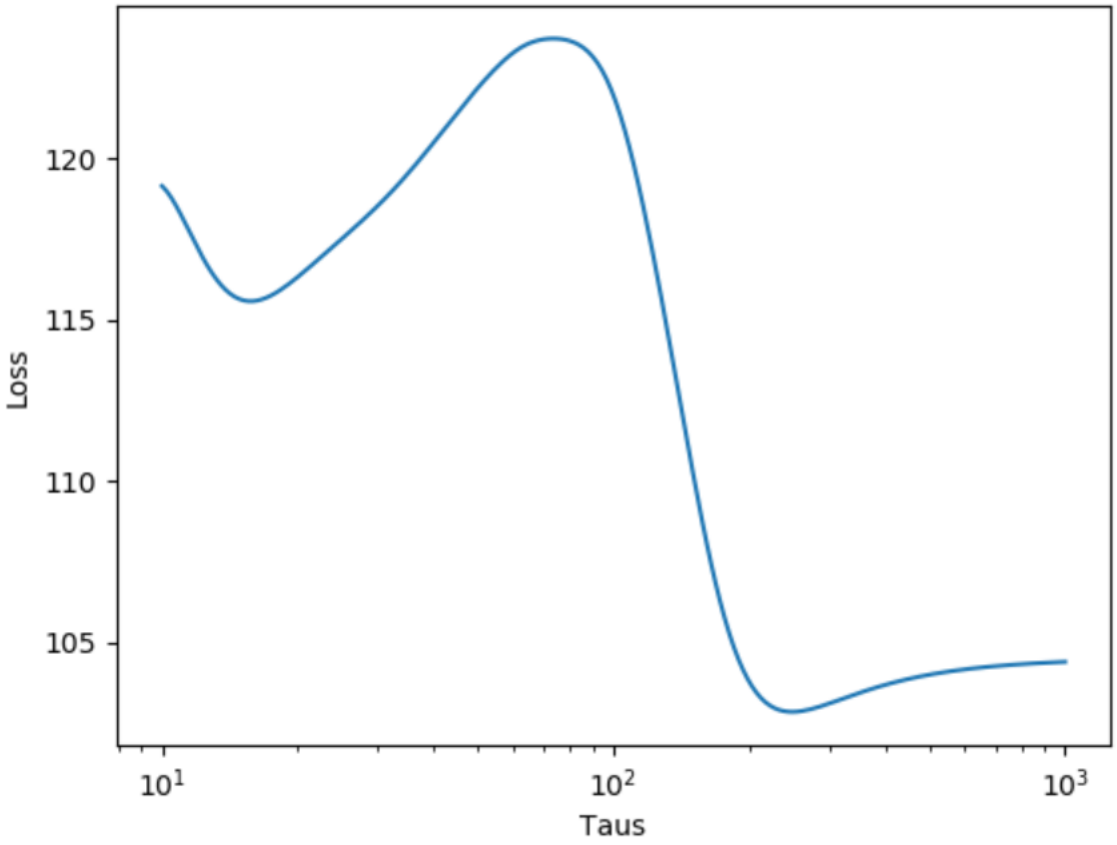
When running the code multiple times, the results do not change as the calculations performed are the same and no result is randomized. For example the weights are not randomized but rather all set to 0.

4 Locally Weighted Regression

4.a

4.b

4.c



4.d

When Tau goes to infinity, the algorithm would behave similarly to linear regression wherein the loss should increase. On the other hand, as Tau goes to 0 the loss should decrease as only as further away points have less weighting. This does not happen in the graph above.