1 Comparing Different Loss Functions

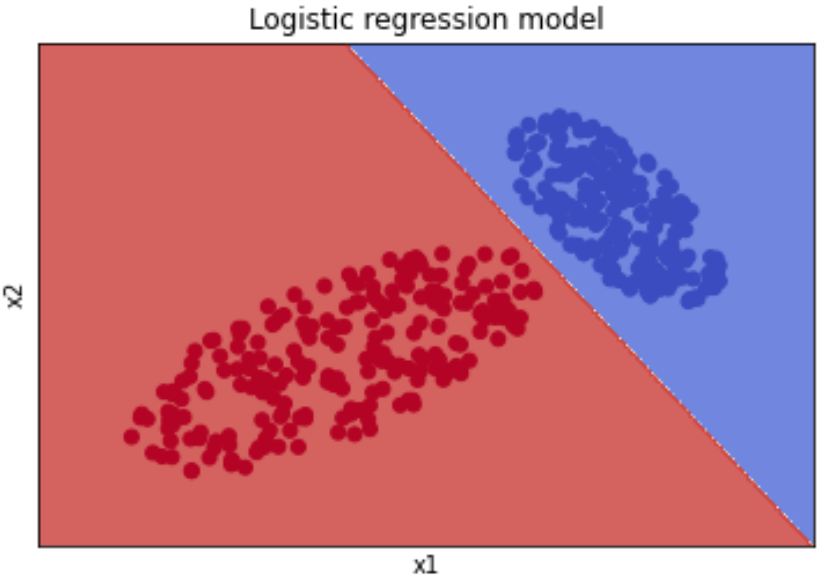
Question 1 Code

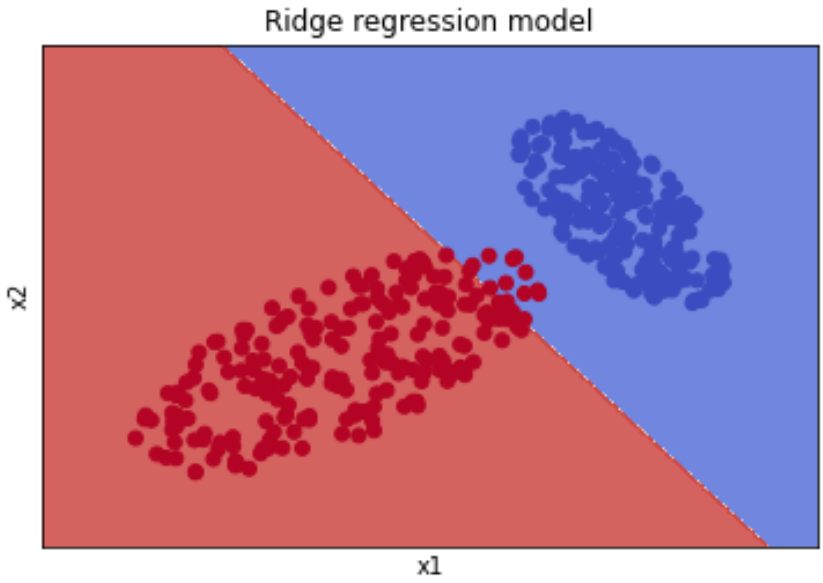
Problem A

Squared loss is often a terrible choice of loss function to train on for classification problems since there can be data points with high error simply because they are far from the decision boundary even if they are classified properly. So, we can imagine how these correctly classified data points will impact the decision boundary due to their relatively high errors due to distance. Thus, using squared loss in the case of classification can be greatly inefficient.

Problem B

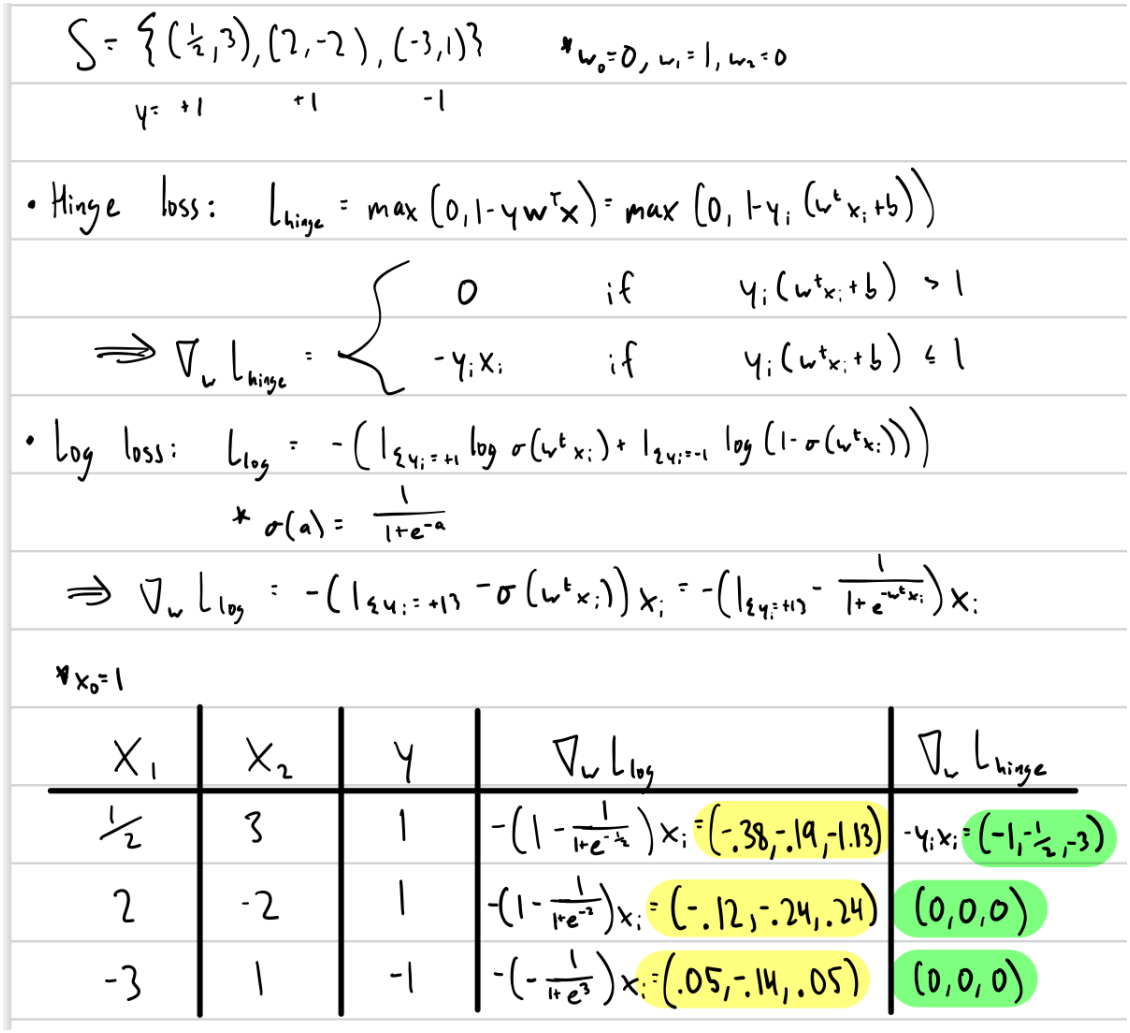
Using the following plots, we can clearly see the differences in the decision boundaries learned using the different loss functions. With the logistic regression model, the decision boundary correctly classifies all of the data points. On the other hand, the decision boundary for the ridge regression model misclassifies a small portion of the "red" data points. These results are explained by the fact that logistic regression uses log loss while ridge regression using squared loss. Therefore, we can imagine how the smaller $x1$ and $x2$ values are considered to have a much higher error with the squared loss function relative to the error assigned with the log loss function. Thus, the use of the squared loss results in the greater shift of the decision boundary such that some of the red data points become misclassified.





Problem C

See work below:



Problem D

We can see that the hinge loss is only nonzero (and greater than log loss from the previous problem) when $y_i(w^t x_i) \leq 1$. The gradients for hinge loss converge to 0 when $y_i(w^t x_i) > 1$ which is also equivalent to $|x_1| > 1$ in this case. Furthermore, the gradients for log loss converge to 0 when $w^t x_i = x_1 \to \infty$ (if $y_i = 1$) or when $w^t x_i = x_1 \to -\infty$ (if $y_i = -1$). In other words, the log loss converges to 0 when x_1 is classified properly and its distance to the decision boundary increases.

Training error with hinge loss can be eliminated by scaling the weight vector (w_0, w_1, w_2) appropriately such that all $y_i(w^t x_i) > 1$ while preserving the decision boundary. Training error with log loss can be reduced (but not fully eliminated) if the weight vector (w_0, w_1, w_2) is similarly scaled up appropriately.

Problem E

Its learning objective must be to minimize $L_{hinge} + \lambda||w||^2$ since minimizing only L_{hinge} can be done without changing the decision boundary by scaling up the weight vector. Thus, the additional penalty term allows for the scaling of the weight vector to be restricted such that the decision boundary properly classifies the data and maximizes the margin.

2 Effects of Regularization

Question 2 Code

Problem A

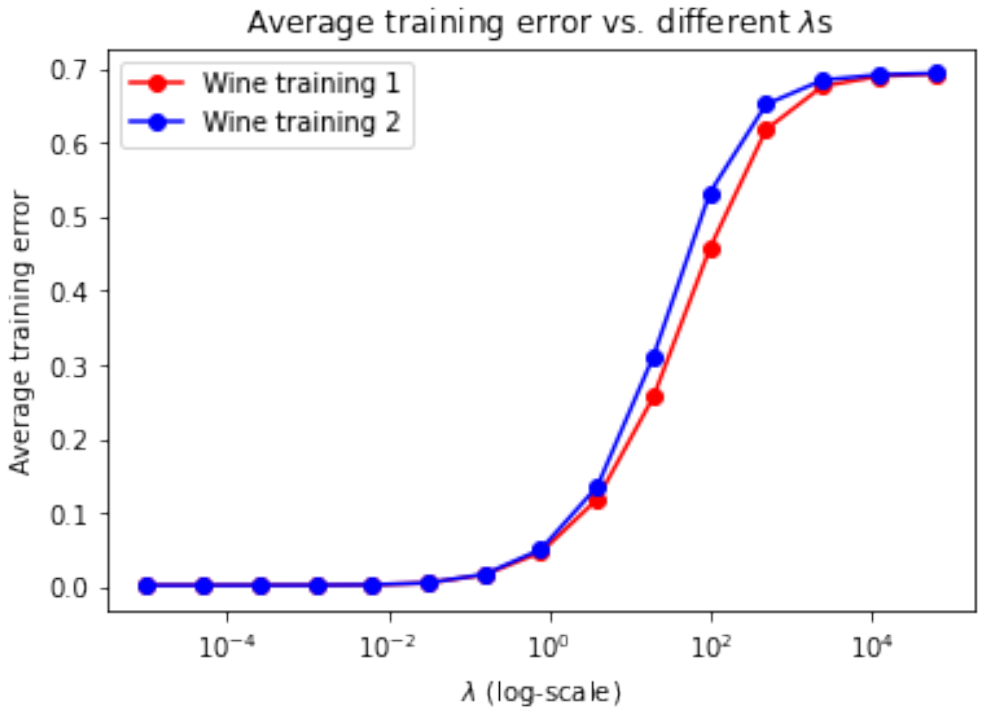
Adding the penalty term will not decrease the training error relative to the unregularized version since this unregularized version already minimizing the training (in-sample) error, so the addition of the penalty term will not decrease the in-sample error any further. Furthermore, particularly in the case when there is over-fitting while training the model, the penalty term can decrease the out-of-sample error since it simplifies and generalizes the model more. However, it is important to note that the addition of the penalty term will not always decrease these out-of-sample errors, particularly if the model is generalized too much, resulting in under-fitting.

Problem B

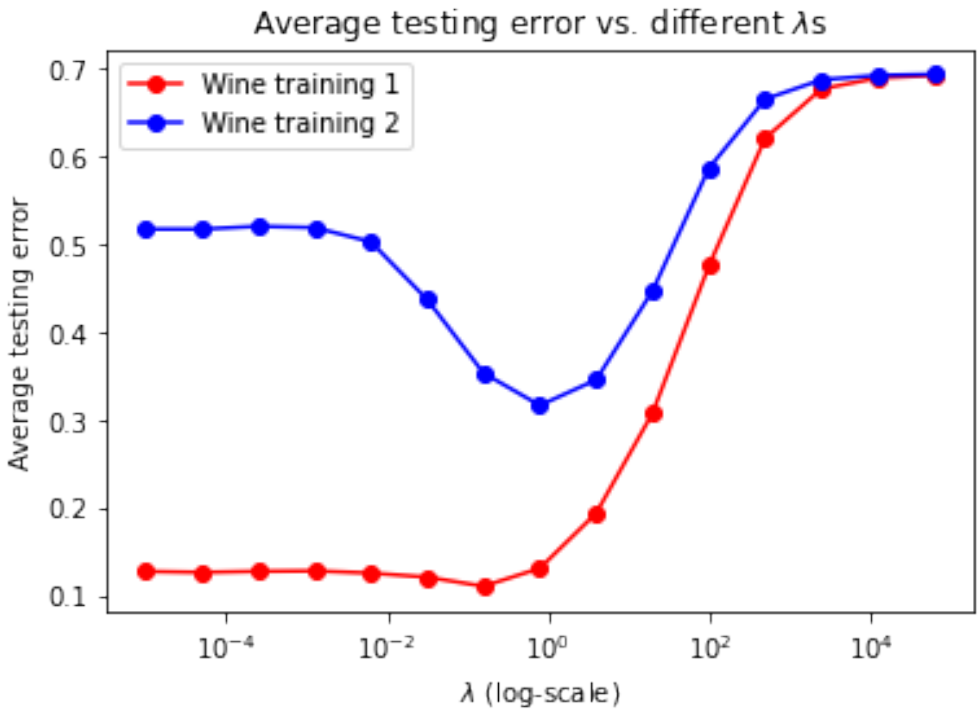
Regularization using the ℓ_0 norm cannot be used directly since it is not continuous. This makes it difficult to optimize the regularization term, and in fact, ℓ_0 is not truly a norm. Thus, it is rarely used.

Problem C

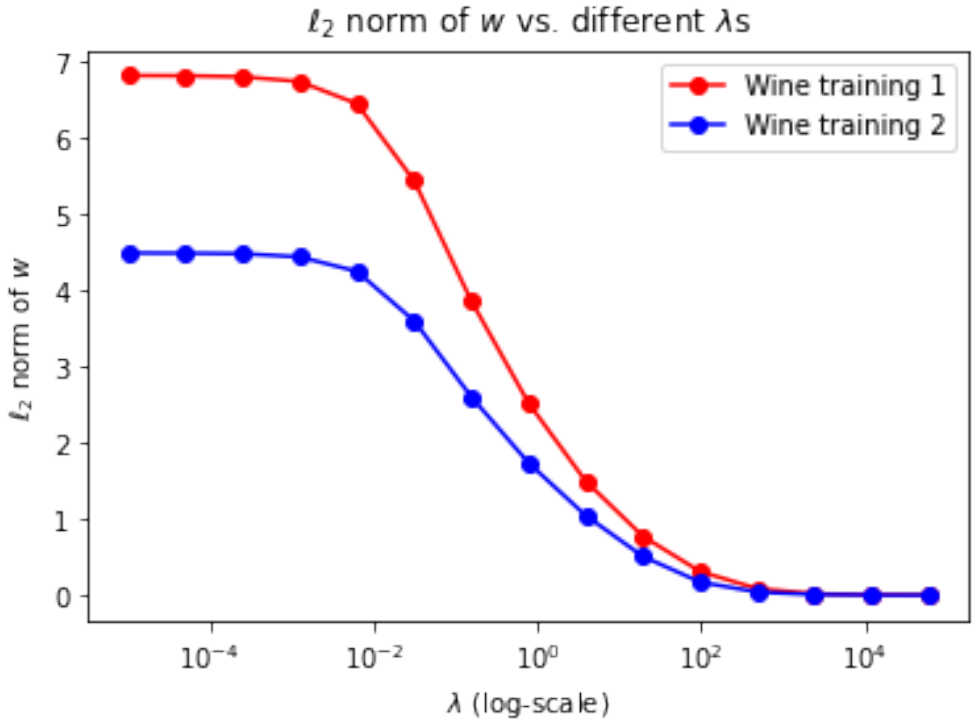
i:



ii:


Average testing error vs. different λs
Wine training 1
Wine training 2
Average testing error
0.7
0.6
0.5
0.4
0.3
0.2
0.1
10^{-4}
10^{-2}
10^{0}
10^{2}
10^{4}
λ (log-scale)

iii:



Problem D

Comparing the training errors, we see that the curves are very similar between winetraining1 and winetraining2. Although, the training error is slightly higher in winetraining2 in the second half of the plot which could be due to its increased vulnerability to regularization since it has fewer data points.

We see that the testing errors are not so similar such that winetraining1 has lower testing error, particularly in the first half of the plot. This is likely due to lower regularization in this range, and since winetraining2 has fewer data points, it would seem that over-fitting has occurred. This also explains that as regularization increases with λ, the testing errors become more similar since it would help to generalize the models and lower the variances.

Problem E

For lower λ values, we see some evidence of over-fitting due to lower training error and higher testing error which makes sense because regularization is low in this range. Although, this over-fitting is not extreme. On the other hand, with higher λ values, there is some evidence for under-fitting as training error increases as the complexity decreases with rising regularization. In the middle range, right before $\lambda = 10^0$, we see a slight dip in the testing error while the training error is still low. This appears to be the ideal λ value for winetraining1.

Problem F

As λ decreases, the ℓ_2 norm of $\mathbf{w}$ decreases for winetraining1. This is because as the regularization increases, we can imagine an increase to the minimization of the regularization term, implying a decrease in the magnitude of $\mathbf{w}$.

Problem G

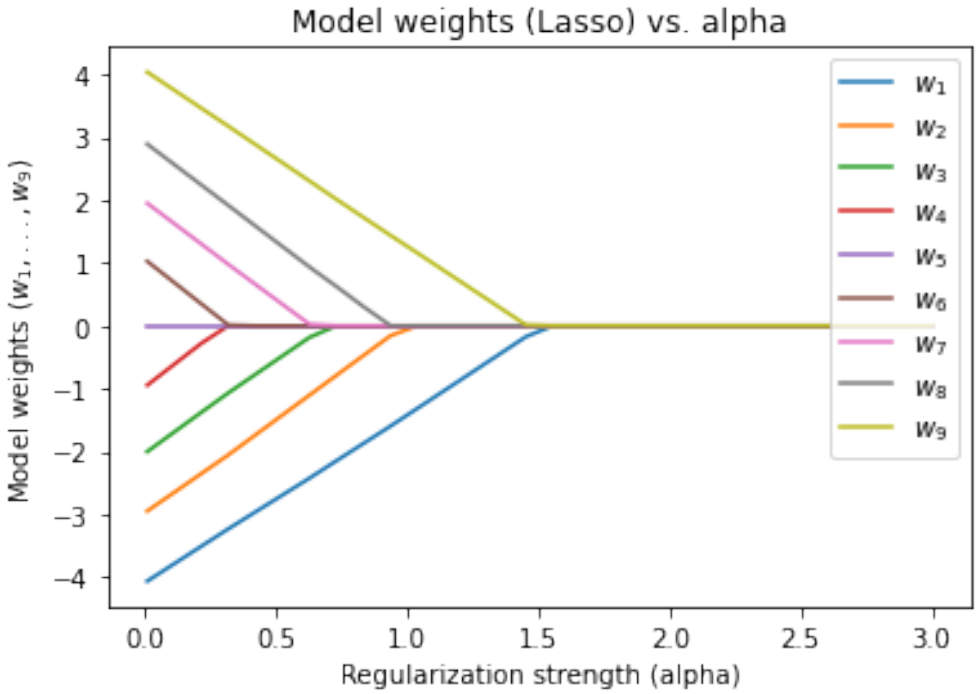
Training with winetraining2, it seems that the ideal λ value is approximately 10^0 since this is where the testing error is at its lowest along with a low training error. Lower λ values mean that the model has more over-fitting, and higher λ values would give us under-fitting.

3 Lasso vs. Ridge Regularization

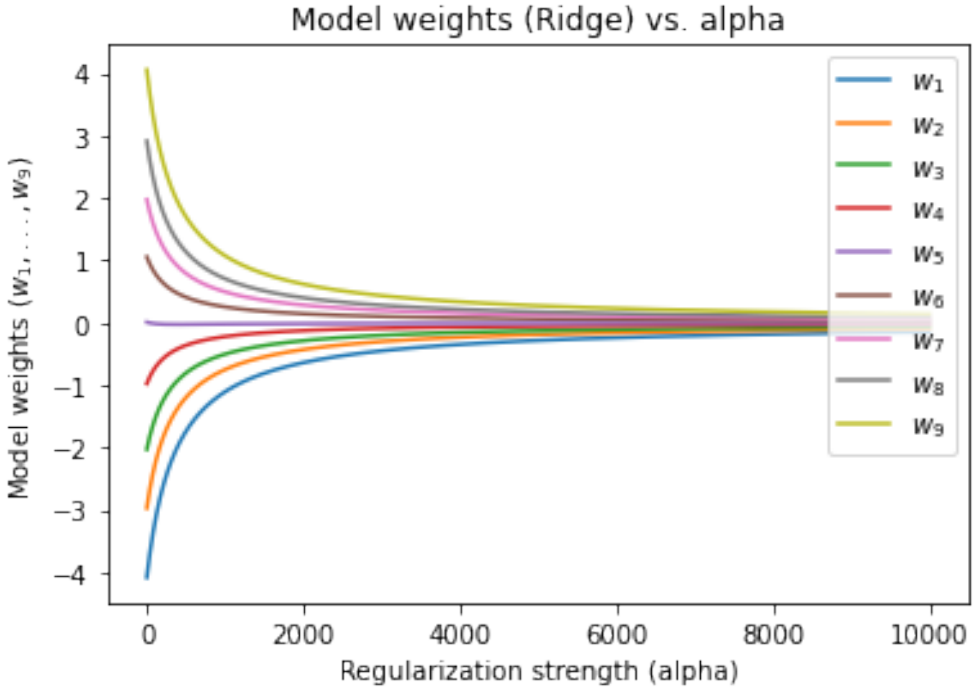
Question 3 Code

Problem A

i:



ii:



iii:

With Lasso regression, the number of model weights that are exactly zero increases as the regularization strength increases such that all of them become zero once $\alpha > 1.5$. With Ridge regression, the model weights get closer to zero as the regularization strength increases, but it does not appear that any of them become exactly zero.

Problem B

i:

$$\nabla_w \left(\|y - Xw\|^2 + \lambda \|w\|_1 \right)$$

$$0 = -2X^T(y - Xw) + \lambda$$

$$0 = -2X^Ty + 2X^TXw + \lambda$$

$$\Rightarrow w = \frac{2X^Ty - \lambda}{2X^TX}$$

ii:

Yes, this value exists.

$$\Rightarrow 2X^Ty - \lambda = 0 \Rightarrow \lambda = \|2X^Ty\|$$

Problem C

i:

$$\nabla_w \left(\|y - Xw\|^2 + \lambda \|w\|_2^2 \right)$$

$$0 = -2X^T(y - Xw) + 2\lambda w$$

$$0 = -2X^T y + 2X^T X w + 2\lambda w$$

$$0 = -X^T y + X^T X w + \lambda w$$

$$\Rightarrow w = \frac{X^T y}{X^T X + \lambda I}$$

ii:

Suppose $w \neq 0$ when $\lambda = 0$.

$\Rightarrow w = \frac{X^T y}{X^T X} \neq 0 \Rightarrow X \neq 0, y \neq 0$

$\Rightarrow$ No value of $\lambda > 0$ can make $w = 0$ true

* can't make $\frac{1}{X^T X + \lambda I} = 0$ true

$\Rightarrow$ Confirmed in A.ii since weights are never exactly 0