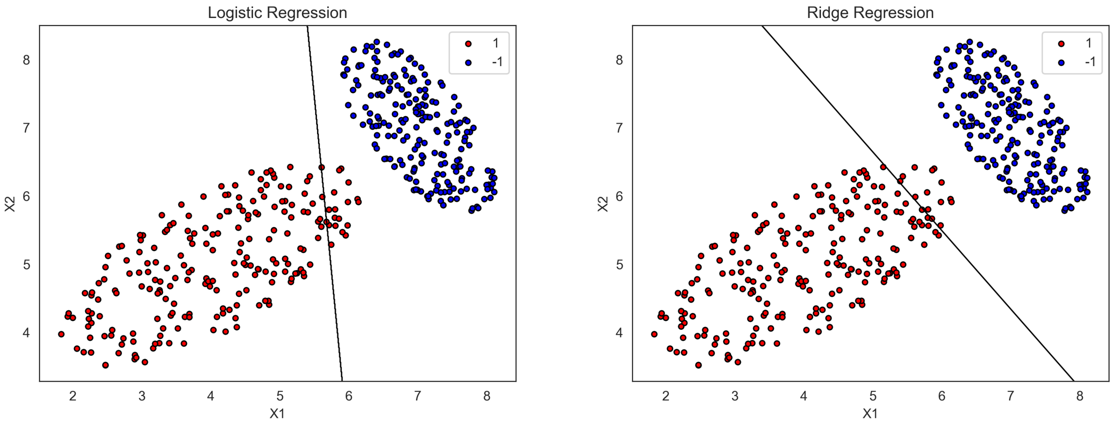
1. A. Squared loss tends to be very bad for classification because we only have two target values $y \in \{-1, +1\}$ whereas the predicted value $w^T x_i$ can be any real value. Therefore, it is very unlikely that the regression will give $y = \pm 1$ and the squared error will not capture what we're looking for.

B. The main qualitative difference between the two separating boundaries is the slope of the line. The linear fit seems to have the line more perpendicular to the data while the logistic regression boundary is more steep.

An explanation is that for logistic regression, cross-entropy error minimization is equivalent to likelihood-maximization, where the target values are interpreted as probabilities. Clearly, the MLE in the data above is a step function dividing the classes. However, in a step function, the weight $||w|| \to \infty$, and with regularzation, we must bound the magnitude of the weights. For the regularization parameter that scikit-learn uses, the boundary turns out to be this.

As we said before in part A, squared-error minimization is a poor choice for classification, and so it happens that the boundary does not separate our separable data. However, due to the symmetry of problem, the ridge regression keeps the boundary line approximately perpendicular to the data points.



(a) Cross-Entropy Error **(b)** Squared Error

C. We have (with regularization)

$$\nabla_w L_{\text{log}} = \frac{1}{2}\lambda \mathbf{w} - \frac{y\mathbf{x}}{1 + e^{y\mathbf{w}^T\mathbf{x}}}, \qquad \nabla_w L_{\text{hinge}} = \frac{1}{2}\lambda \mathbf{w} + \begin{cases} 0 & y\mathbf{w}^T\mathbf{x} > 1 \\ -y\mathbf{x} & y\mathbf{w}^T\mathbf{x} \leq 1 \end{cases}$$

For the points in S, we have the table shown below.

S	$\nabla_w L_{\text{log}}$	$\nabla_w L_{\text{hinge}}$
(1/2, 3)	(-0.38, -0.19, -1.13)	(-1, -0.5, -3)
(2, -2)	(-0.12, -0.24, 0.24)	(0, 0, 0)
(-3, 1)	(0.047, -0.14, 0.047)	(0, 0, 0)

D. For the log loss, without regularization ($\lambda = 0$), we can never get a case where $\nabla_w L_{\text{log}} = 0$ unless the weights go to infinity, in which case the weights don't converge.

For the hinge loss, we can get a case where $\nabla_w L_{\text{hinge}} = 0$ if all the points satisfy $y\mathbf{w}^T\mathbf{x} > 1$. If we get a point that doesn't satisfy the condition, then the next iteration yields

$$\mathbf{w} \to \mathbf{w} + \eta y \mathbf{x} \implies y\mathbf{w}^T\mathbf{x}' \to y(\mathbf{w} + \eta y\mathbf{x})^T\mathbf{x}' = y\mathbf{w}^T\mathbf{x}' + \eta\mathbf{x}^T\mathbf{x}'$$

For the corrected point where $\mathbf{x}' = \mathbf{x}$, we see that $\eta\mathbf{x}^T\mathbf{x} > 0$ so that our margin $y\mathbf{w}^T x$ increases and at some point surpasses 1 so that our hinge gradient is zero.

Thus, for a linearly separable dataset, the cross-entropy error will never be zero because the weights have to diverge. For hinge-loss, the training error could become zero.

E. SVM is a "maximum margin" classifier because the motivation for SVM is to maximize the margin between the classes. By normalizing $|\mathbf{w}^T\mathbf{x}| = y\mathbf{w}^T\mathbf{x} = 1$ at the margin, this problem becomes equivalent to maximizing $1/|w|$. This is equivalent to minimizing

$$\frac{1}{2}\mathbf{w}^T\mathbf{w} \qquad \text{subject to} \qquad y\mathbf{w}^T\mathbf{x} \geq 1.$$

Using Lagrange multipliers, we want to minimize the following Lagrangian,

$$\mathcal{L} = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_n \alpha_n(y_n\mathbf{w}^T\mathbf{x}_n - 1).$$

If we impose soft constraints where we can violate the margin at a certain cost, we have

$$y_n\mathbf{w}^T\mathbf{x}_n \geq 1 - \xi_n, \qquad \xi_n \geq 0$$

where ξ_i is the margin violation. We impose a cost, C, where we now impose the minimization

$$\mathcal{L} = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_n \xi_n - \sum_n \alpha_n(y_n\mathbf{w}^T\mathbf{x}_n - 1 + \xi_n) - \sum_n \beta_n\xi_n$$

We see that we can interpret the above Lagrangian as trying to minimize the hinge loss ξ_n with respect to some constraints as well as adding a regularization on $\mathbf{w}$ with $\lambda = 1$.

2. A. No, adding a penalty term cannot decrease in-sample error because in-sample error is always directly minimized. Adding a penalty term, however, does not always decrease out-of-sample error. A large penalty can cause underfitting which can hurt out-of-sample error.

B. We don't use ℓ_0 regularization because

$$||w||_0 = \sum_d 1_{[w_d \neq 0]}$$

is not continuous and thus is difficult to computationally optimize. Also, ℓ_1 is sparse enough and continuous.

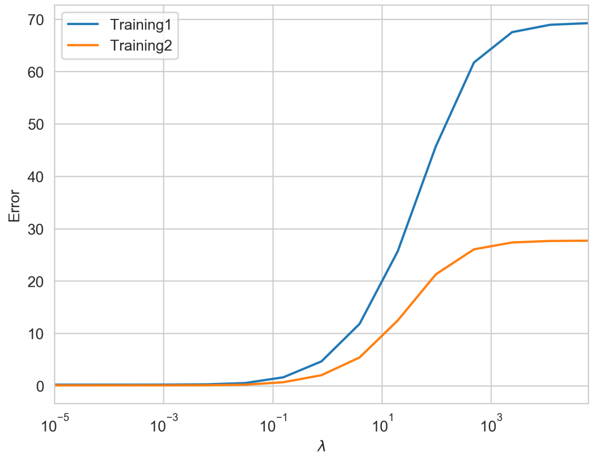


Figure 2: In-sample error for different λ's.

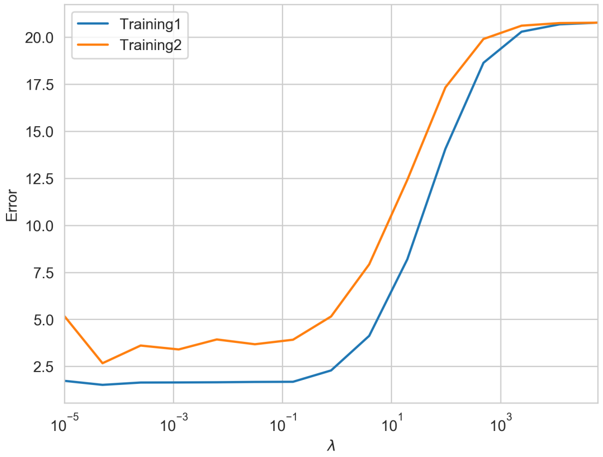


Figure 3: Out-of-sample error for different λ's.

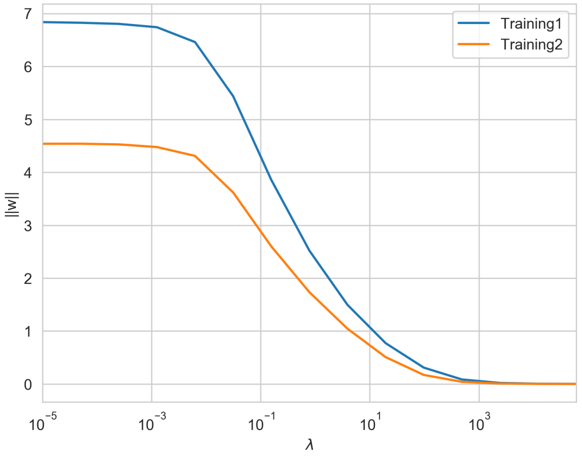


Figure 4: L2-Norm $||w||$ for different λ's.

C. We can see how different λ's affect our training in the following three figures.

D. The in-sample error is smaller for the subset because there are less points to fit with the logistic model. However, out-of-sample error is larger for the subset because it has less training samples, our estimation of the weights that minimizes out-of-sample has more variance so that our final weight is further from the ideal weight that minimizes the out-of-sample error. Finally, it is clear that the weight norm decrease with a higher λ because the regularization term dominates as λ increases and then minimizing the loss function essentially becomes minimizing the regularization term. The subset yields a smaller $||w||$ because the loss function is more dominated by the regularization term since the error is summed over less samples, so the training algorithm puts more emphasis on decreasing the regularization term.

E. There is not much of a case of overfitting here. As we reach in the realm of high $\lambda \gg 0.1$, we see that the in-sample error increases dramatically (obviously), but the out-of-sample error also increases dramatically. This is a clear sign of underfitting, implying that high-regularization term restricts the model of the complexity capture in the actual target function.

F. The minimum value of E_{out} for the Training2 data is $\lambda = 5 \times 10^{-5}$, so I would choose that λ.

3. A. i. We see in Figure 5 how the discontinuity and linearity of the L1 norm forces the weights to zero.

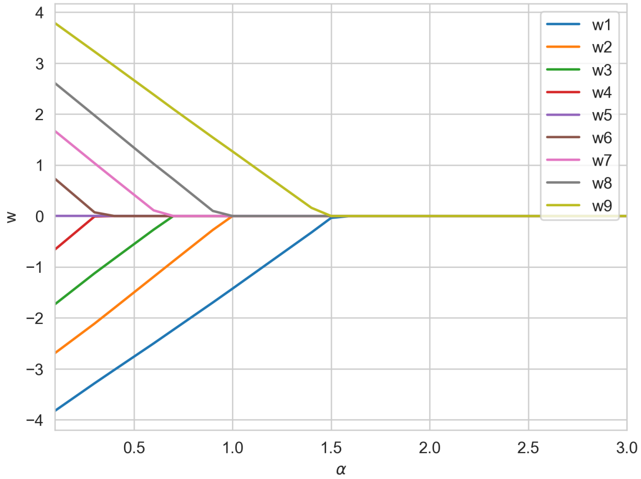


Figure 5: Weights versus regularization parameter α. For Lasso regularization, the weights quickly die to zero, creating a sparse weight vector.

ii. We see in Figure 6 that weights don't go to zero, but the norm shrinks to zero as $\alpha \to 0$.

iii. As said in parts i and ii, for Lasso, as α increases, the number of model weights that are exactly zero increases. In ridge regression, the weights don't hit exactly zero, but only get smaller as α increases, as expected.

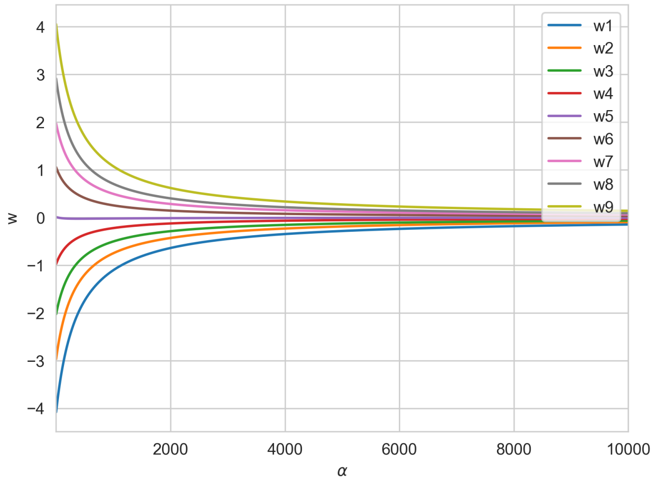


Figure 6: Weights verus regularization parameter α. For ridge regularization, the weights shrink, but never reach zero, as α increases due to the smoothness of $\mathbf{w}^T\mathbf{w}$.

B. i. In this case, w is just a scalar, so we need to calculate

$$\arg\min_w L = \arg\min_w \left(||\mathbf{y} - w\mathbf{x}||^2 + \lambda|w|\right)$$

Differentiating with respect to w gives

$$-2\mathbf{x}^T\mathbf{y} + 2w\mathbf{x}^T\mathbf{x} + \lambda\frac{|w|}{w} = 0$$

The $w > 0$ solution is

$$w = \frac{1}{2\mathbf{x}^T\mathbf{x}}\left(2\mathbf{x}^T\mathbf{y} - \lambda\right).$$

The $w < 0$ solution is

$$w = \frac{1}{2\mathbf{x}^T\mathbf{x}}\left(2\mathbf{x}^T\mathbf{y} + \lambda\right),$$

and the $w = 0$ solution is, of course,

$$w = 0.$$

We have to decide which one minimizes the Lasso error function. In the $w < 0$ case, we have

$$\begin{aligned}
L_{w>0} &= \left|\left|\mathbf{y} - \frac{\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x}}\mathbf{x} + \frac{\lambda\mathbf{x}}{2\mathbf{x}^T\mathbf{x}}\right|\right|^2 + \lambda\left|\frac{\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x}} - \frac{\lambda}{2\mathbf{x}^T\mathbf{x}}\right| \\
&= \mathbf{y}^T\mathbf{y} + \frac{(\mathbf{x}^T\mathbf{y})^2}{\mathbf{x}^T\mathbf{x}} + \frac{\lambda^2}{4\mathbf{x}^T\mathbf{x}} - 2\frac{(\mathbf{x}^T\mathbf{y})^2}{\mathbf{x}^T\mathbf{x}} + \frac{\lambda\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x}} - \frac{\lambda\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x}} \\
&\quad + \lambda\left|\frac{\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x}} - \frac{\lambda}{2\mathbf{x}^T\mathbf{x}}\right| \\
&= \mathbf{y}^T\mathbf{y} - \frac{(\mathbf{x}^T\mathbf{y})^2}{\mathbf{x}^T\mathbf{x}} + \frac{\lambda^2}{4\mathbf{x}^T\mathbf{x}} + \lambda\left|\frac{\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x}} - \frac{\lambda}{2\mathbf{x}^T\mathbf{x}}\right|
\end{aligned}$$

If $2\mathbf{x}^T\mathbf{y} > \lambda$, then

$$L_{w>0,2\mathbf{x}^T\mathbf{y}>\lambda} = \mathbf{y}^T\mathbf{y} - \frac{(\mathbf{x}^T\mathbf{y})^2}{\mathbf{x}^T\mathbf{x}} + \frac{\lambda\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x}} - \frac{\lambda^2}{4\mathbf{x}^T\mathbf{x}} = \mathbf{y}^T\mathbf{y} - \frac{(2\mathbf{x}^T\mathbf{y} - \lambda)^2}{4\mathbf{x}^T\mathbf{x}}$$

On the other hand,

$$L_{w>0,2\mathbf{x}^T\mathbf{y}<\lambda} = \mathbf{y}^T\mathbf{y} - \frac{(\mathbf{x}^T\mathbf{y})^2}{\mathbf{x}^T\mathbf{x}} - \frac{\lambda\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x}} + \frac{3\lambda^2}{4\mathbf{x}^T\mathbf{x}}$$

Similarly,

$$L_{w<0} = \mathbf{y}^T\mathbf{y} - \frac{(\mathbf{x}^T\mathbf{y})^2}{\mathbf{x}^T\mathbf{x}} + \frac{\lambda^2}{4\mathbf{x}^T\mathbf{x}} + \lambda\left|\frac{\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x}} + \frac{\lambda}{2\mathbf{x}^T\mathbf{x}}\right|.$$

So we have for $2\mathbf{x}^T\mathbf{y} + \lambda < 0$

$$L_{w<0,2\mathbf{x}^T\mathbf{y}<-\lambda} = \mathbf{y}^T\mathbf{y} - \frac{(2\mathbf{x}^T\mathbf{y} + \lambda)^2}{4\mathbf{x}^T\mathbf{x}}$$

$$L_{w<0,2\mathbf{x}^T\mathbf{y}>-\lambda} = \mathbf{y}^T\mathbf{y} - \frac{(\mathbf{x}^T\mathbf{y})^2}{\mathbf{x}^T\mathbf{x}} + \frac{\lambda\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x}} + \frac{3\lambda^2}{4\mathbf{x}^T\mathbf{x}}$$

And finally,

$$L_{w=0} = \mathbf{y}^T\mathbf{y}.$$

Now if $2\mathbf{x}^T\mathbf{y} > \lambda$, then $L_{w>0,2\mathbf{x}^T\mathbf{y}>\lambda}$ definitely has the lowest error. If $2\mathbf{x}^T\mathbf{y} < -\lambda$, then $L_{w<0,2\mathbf{x}^T\mathbf{y}<-\lambda}$ has the lowest error. What happens in the case of $-\lambda < 2\mathbf{x}^T\mathbf{y} < \lambda$? The $w > 0$ solution yields

$$\mathbf{y}^T\mathbf{y} - \frac{(\mathbf{x}^T\mathbf{y})^2}{\mathbf{x}^T\mathbf{x}} - \frac{\lambda\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x}} + \frac{3\lambda^2}{4\mathbf{x}^T\mathbf{x}} > \mathbf{y}^T\mathbf{y} - \frac{\lambda^2 + 2\lambda^2 + 3\lambda^2}{4\mathbf{x}^T\mathbf{x}} = \mathbf{y}^T\mathbf{y}$$

and the $w < 0$ solution yields

$$\mathbf{y}^T\mathbf{y} - \frac{(\mathbf{x}^T\mathbf{y})^2}{\mathbf{x}^T\mathbf{x}} + \frac{\lambda\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x}} + \frac{3\lambda^2}{4\mathbf{x}^T\mathbf{x}} > \mathbf{y}^T\mathbf{y} - \frac{\lambda^2 - 2\lambda^2 + 3\lambda^2}{4\mathbf{x}^T\mathbf{x}} = \mathbf{y}^T\mathbf{y}.$$

So in this realm, the $L_{w=0}$ solution wins. Note the continuity of the solutions in $2\mathbf{x}^T\mathbf{y}$. In summary, we have

$$\begin{aligned} w &= \frac{1}{2\mathbf{x}^T\mathbf{x}}(2\mathbf{x}^T\mathbf{y} - \lambda) \qquad \text{if} \qquad 2\mathbf{x}^T\mathbf{y} > \lambda \\ w &= \frac{1}{2\mathbf{x}^T\mathbf{x}}(2\mathbf{x}^T\mathbf{y} + \lambda) \qquad \text{if} \qquad 2\mathbf{x}^T\mathbf{y} < -\lambda \\ w &= 0 \qquad \text{if} \qquad -\lambda \leq 2\mathbf{x}^T\mathbf{y} \leq \lambda. \end{aligned}$$

ii. And of course, the smallest value of λ such that $w = 0$ is $\lambda = |2\mathbf{x}^T\mathbf{y}|$.

iii. Differentiating with respect to $\mathbf{w}$ and setting to zero yields

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

so

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X} + \lambda I)^{-1}\mathbf{X}^T\mathbf{y}.$$

iv. In one-dimension, our solution becomes

$$w = \frac{\mathbf{x}^T\mathbf{y}}{\mathbf{x}^T\mathbf{x} + \lambda}.$$

This is not zero unless $\mathbf{x}^T\mathbf{y} = 0$ (which is impossibly rare). Nevertheless, choosing λ won't affect whether it's zero or not.