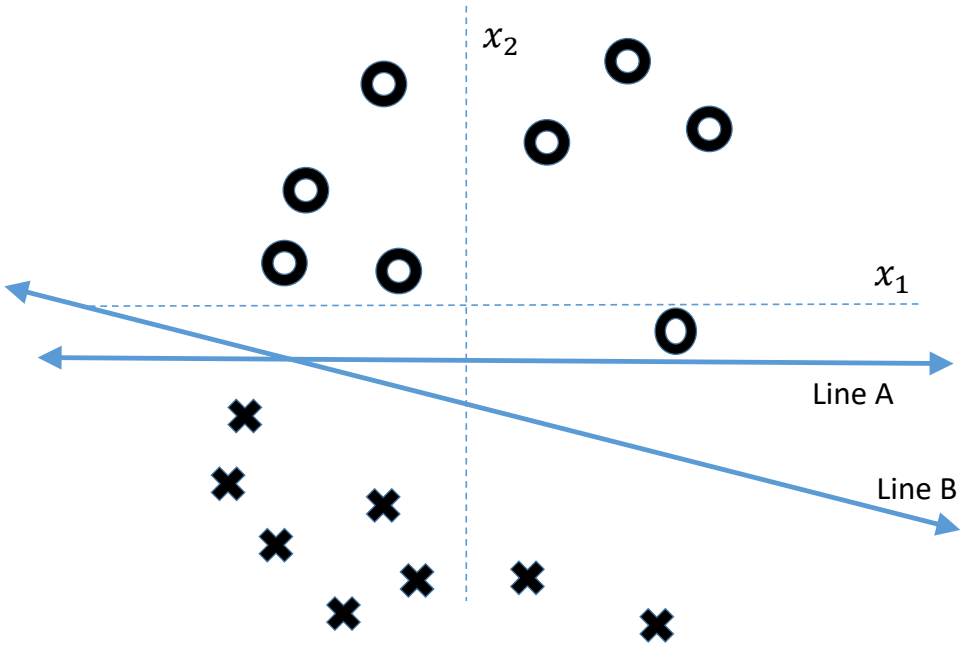
Section 05: Solutions

1. Logistic Regression

Let's think about what happens with linearly separable data in logistic regression.

Consider the following training set:



Each data point has two features – one shown on the horizontal axis, and the second on the vertical axis. Each point is in one of two classes – 'X' is class 1, 'O' is class -1. Our model is logistic regression:

$$\Pr(y = 1|x, w) = \frac{1}{1 + \exp(-w_0 - w_1 x[1] - w_2 x[2])}$$

We've shown two lines that might have been the result of training on our training set.

(a) What are some possible values of w to produce line A and line B. Line A is horizontal with an intercept of -1. Line B has a slope of $-1/2$ and an intercept of -2 on the vertical axis.

Solution:

Recall that the lines separating the positive/negative samples are the sets of points where we're setting $\Pr[y = 1|x, w] = 1/2$, so we want to define a w such that $w_0 + w^T x = 0$ on the line, and so that our estimate of the probability of a sample being 1 increases as we go further on the side of the line where the 1 samples are.

Line A: we want $w_0 + w^T x$ to be positive if and only if we're below the line (because we want our estimate that $\Pr(y = 1)$ to approach 1, and therefore $-(w_0 - w_1 x[1] - w_2 x[2])$ to approach $-\infty$ as we go farther into the area where all the positive examples are.) To accomplish this, note that the dot product with a vector perpendicular to the line gives which side of the line a point is on. A perpendicular vector is to A is $[0, -1]$ (note that $[0, 1]$ would not work, as it would cause $\Pr(y = 1)$ to go to 0, not 1). We now need to calculate w_0

Note that $w_0 + w^T x$ should be 0 whenever we input a point x on line A so $w_0 + x_1 w_1 + -1 \cdot w_2 = 0$ Plugging in $x_1 = 0$, we get $w_2 = w_0$, so we can set $w = [-1, 0, -1]$.

Line B: a perpendicular vector to a line of slope $-1/2$ is $[-1, -2]$. Plugging in the fact that at $x = (0, -2)$ we should get 0 we have: $w_0 + 0(-1) + (-2)(-2) = 0$ so $w_0 = -4$, and our predictor is $w = [-4, -1, -2]$

(b) Show that for the w you found in part a, that $2w$ still corresponds to the same separating hyperplane (i.e. to the same lines A and B)

Solution:

Since we know $w_0 + w^T x = 0$, on the line, we have $2w_0 + (2w)^T x = 2(w_0 + 2^T x) = 2 \cdot 0 = 0$.

(c) In lecture, we said a common loss function for this problem is $J(w) = \sum_{i=1}^{n} \log\left(1 + \exp(-y_i x_i^T w)\right)$ If we try to minimize this error, the w we found in part a. won't actually be $\hat{w} = \arg\min \sum_{i=1}^{n} \log\left(1 + \exp(-y_i x_i^T w)\right)$ – why? (think about part b)

Solution:

If we double either of the w vectors we had above, we'll be predicting the same separating hyperplane, (i.e. we're just doubling the magnitude of the perpendicular vector we're taking the dot product with) since we're getting every point correct, our loss only goes down as we scale w up. In effect, our loss rewards our model for being "confident" when it is correct. Since we're always correct, we can increase our "confidence" in our own predictions arbitrarily and get rewarded more and more on the training set.

(d) Why is this an issue, and how do we fix it?

Solution:

Our w will grow arbitrarily large in magnitude. This is a problem for two reasons, first whatever algorithm we're using to find w may not converge. If we do somehow get a finite vector for an answer, it's likely that we'll be heavily overfitting to our data – we'll have high confidence on samples close to whatever line we've chosen, but we don't have any training data around those points, so shouldn't be confident (and will likely suffer in our test error as a result).

We can fix the problem with regularization.

(e) Suppose we regularized our error, by adding the term $\lambda(|w_1| + |w_2|)$ to our objective (i.e. L1 regularization). For large values of λ which of the lines do you expect to have smaller objective value (i.e. smaller error plus regularization penalty)? (We're looking for intuition, not a calculation. Hint: why do we usually use L1 regularization?)

Solution:

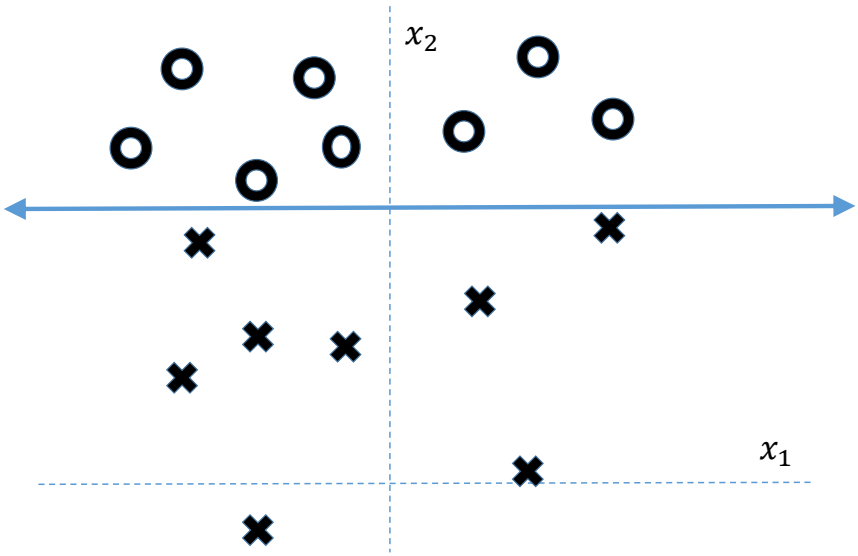
The horizontal line seems like a better choice – it ignores one of the features (since the line is horizontal, it makes the same prediction regardless of the value of the feature on the horizontal axis. I.e. w_1 is 0.). Sparse vectors are usually selected by L1 regularization. Notice that this solves the problem from the previous part. We might be able to decrease $J(w)$ by continually scaling up w, but when we scale w up, the regularization penalty also scales up by λ, so we won't be rewarded for arbitrary scaling.

(f) We didn't regularize by w_0. Give an example of a data set where regularizing with $\lambda(|w_0| + |w_1| + |w_2|)$ would

lead to much worse behavior than regularizing by $\lambda(|w_1| + |w_2|)$.

Solution:

Consider the following dataset:



There is a w which will classify every point correctly, but it has a very large intercept. If we regularize by w_0, we will be forced to decrease w_0, but if we decrease w_0 then we will start misclassifying points.

(g) Suppose you've let λ grow so large that w_1 and w_2 have been set to 0. What kind(s) of functions are we now capable of modeling? What value of w_0 will minimize the error?

Solution:

With w_1 and w_2 equal to 0, we no longer have any contribution from x. We are just predicting a single probability p for all points. To minimize error, we should set w_0 so that the output probability is the fraction of training set points with $y = 1$. Since we have the same number of points in each of our classes, that means $w_0 = 0$.

2. Solving Problems

You're preparing for a meeting with a client that you just finished making a regression model for, and the client does NOT seem happy. You have the ability to do any of the following techniques to try to improve your model.

a. Find a larger training set.

b. Try using a more complex hypothesis class

c. Try L1 regularization

d. Change the set of features

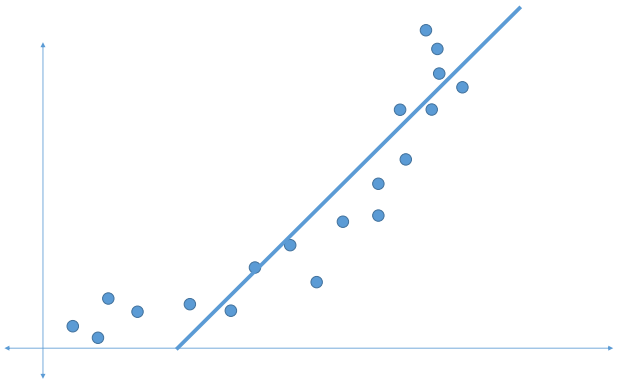
For each of the following complaints your client could give you, which of the above techniques is most likely to make them happy?

(a) "I can't tell what this thing is doing. Is the number of bathrooms important for the selling price or not?"

Solution:

When you can't tell which features are important, L1 regularization is a step in the right direction. This doesn't completely solve the client's issue (maybe the regularization will select a highly correlated feature instead) but it is a start. [c]

(b) "Look at this plot, do these predictions look accurate to you?



Solution:

No. They don't. The plot shows a non-linear relationship between the features and the output (it might be quadratic or exponential), and we've learned a linear one. This is an issue with our hypothesis class – we need to expand it to learn the true function. [b]

(c) "Can you really recognize handwritten digits with just linear combinations of pixel values? It doesn't seem like we're getting low error here."

Solution:

This is a problem with the feature set. Single pixel values aren't a meaningful way to identify what number someone wrote – the data just won't be nicely separable in this space. This will "show up" as high irreducible error. No matter how much larger we make our training set, we're not going to learn a good function. If we change the features, we may be able to move the points into a space where we can separate them. [d]

(d) "We trained your model on two different subsets of the test set and got very very different predictions from each half."

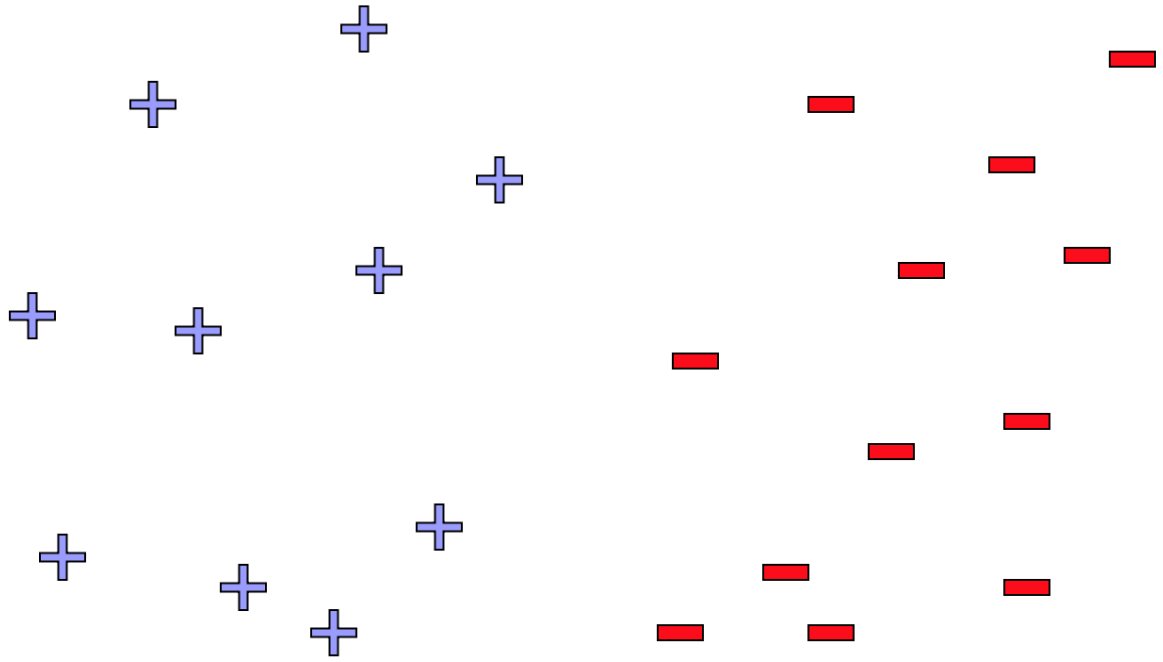
Solution:

Sounds like our model is too high-variance for the number of data points we're operating on. Hopefully if we increase the size of the training set, our variance will drop [a]

3. More Practice

3.1. More Logistic Practice

Let's think about things we need to be careful about in machine learning. We will use an important concepts in machine learning to solve this problem. Consider the following picture obtained from lecture,



where each data points are i.i.d. of the form $(x_i, y_i)_{i=1}^n$ and $x_i \in R^d$, $y_i \in \{-1, 1\}$. Intuitively, you can think of this data having $d = 2$, and imagine 2 axis draw across this image, which would simplify your thought process to solve this problem.

Notice that we will use some characteristics about these data points to inform us of our decision choices when training our algorithm.

(a) Suppose we have some classification problems to solve for the above data points. Which loss function should we use to train on?

Solution:

$\sum_{i=1}^n log(1 + exp(-y_i(x_i^T w + b))) + \lambda ||w||^2$

(b) Provide justification for this loss function. What is the interpretation of each part of the function? Why is regularization particularly important in this instance?

Solution:

Recall that in our model, we interpreted $\frac{1}{1+exp(-(x_i^T w+b))}$ as our estimate of the "probability" that a data point at x would be part of class 1. As $-(x_i^T w + b))$ increases in magnitude, our estimates get closer to 0 and 1. The function $\log(1+exp(-y_i(x_i^T w+b)))$ gives a constant penalty whenever we predict a probability of $1/2$ for $y_i = 1$. As $|x_i^T w + b|$ increases, our loss grows essentially linearly if we are "wrong" (e.g. we predict a probability of more than $1/2$, but y_i is in class -1), while our loss decreases slowly if we are correct.

A key feature of our objective is the regularization term. Since the data is linearly separable, we must

impose a penalty on very confident models. This is to prevent the weights to going to infinity which would minimize the error without the regularization.

(c) Why would it be bad not to regularize?

Solution:

We cannot make assumption about data we have not seen before, this is a finite set of data which happens to be linearly separable. It might be possible for future data to have negatives in the positive region and vice versa. An infinite weight would make huge error in such predictions if we did not regularize the error.

3.2. True/False, Multiple Choice

(a) Suppose you're using L2 regularization on a least squares objective. Some value λ^* will give you the best test error among all possible λ. You train your model using a $\lambda' << \lambda^*$. Which of the following do you expect to be true about your training error?

a. The training error for λ' will be much smaller than for λ^*

b. The training error for λ' will be much bigger than for λ^*

c. The training error for each will be about the same – we regularize for the effect on test error.

Solution:

a, if we're not regularizing as much as we should, then we are likely to be overfitting to our training set – thus our training error will be smaller with λ'.

(b) The maximum likelihood estimator is always unbiased.

Solution:

False. We've seen an example in class – the MLE for the variance of a Gaussian was biased.

(c) Gradient descent will always converge to a global minimum on a convex function

Solution:

False, if the learning rate is too high, we may diverge.

(d) Why might it be difficult to choose a good learning rate for gradient descent with L1 regularization?

a. The gradient will have many terms, so each update step is slow.

b. Near the minimum, when the regularization term dominates, the gradient might stay at a constant magnitude.

c. The L1 regularization makes the function non-convex, and might introduce local minima.

Solution:

b – If we set the learning rate small enough to avoid bouncing around near the minimum, we risk taking a long time to get anywhere near the minimum.

(e) Suppose f is a convex function, g is a concave function, and h is a linear function (i.e. $h(x+y) = h(x)+h(y)$). For each of the following, say whether the function is concave, convex, or that you can't give a guarantee.

(i) $f - g$ **Solution:**

Convex. Recall that if g is concave, $-g$ is convex. So $f - g$ is the sum of convex functions, which you proved on the homework is convex.

(ii) $f + h$ **Solution:**

Convex. Linear functions are also convex, so this is the sum of convex functions.

(iii) $g + h$ **Solution:**

Concave. A similar proof to the one from the homework will show the sum of concave functions is concave.

(iv) fg **Solution:**

Can't give a guarantee. If $f(x) = x$ and $g(x) = -x$ then $fg = -x^2$ is concave, but if $f(x) = g(x) = x$, $fg = x^2$ is convex. There are also functions like $f = x^2$, $g = (x-2)^2$ where the function is neither concave nor convex.

(f) If you are trying to find the minimum of a convex function and choose a learning rate small enough for convergence, which of the following is true?

a. The function value is decreasing at each iteration for both gradient descent and stochastic gradient descent.

b. The function value is decreasing at each iteration for gradient descent but possibly not for stochastic gradient descent.

c. The function value is decreasing at each iteration for stochastic gradient descent and but may not for gradient descent.

d. The function value could go up and down for both gradient descent and stochastic gradient descent, but you'll eventually arrive at the global minimum.

Solution:

b. Since SGD uses a random step, you might happen to choose a data point that will cause the overall objective to go up. Gradient descent is always stepping in the direction of steepest descent, for the overall error so we always go down. (Note that we assumed we made a good choice of learning rate – if we choose too large of a learning rate, gradient descent can diverge).

3.3. Free Response Questions

(a) Describe some differences between linear and logistic regression.

Solution:

Linear regression is a regression model and outputs a real number, while logistic regression is meant for classification and predicts a probability for each class. Additionally, linear and logistic regression have different loss functions. Linear regression has a closed form solution, while logistic does not.

(b) Why might we prefer convex loss functions over non-convex ones?

Solution:

With convex loss functions, local minima are global minima, so with an appropriate setting of the learning rate, we will converge to the global minimum. With a non-convex loss, we are not even guaranteed to converge to a minimum.

(c) Why would we want to use SGD in practice instead of GD?

Solution:

To compute the average gradient, we need to pass through the entire dataset, which can be expensive (consider Instagram has over 20 billion images). Picking one sample gives us an unbiased estimate of the gradient and can be much cheaper to compute.

(d) Sort the following models by variance of the model class:

(i) Quadratic functions

(ii) Constant functions (i.e. functions that output the same prediction for all data points)

(iii) Linear functions

Solution:

A quadratic fit will have the highest variance, followed by linear models, followed by constants.
In general, the more expressive a model is, the more variance it is likely to have. Since these classes are strictly more expressive than each other, (e.g. any linear function can be expressed in a quadratic model) we can use that to predict the variance.

4. Machine Learning Terms

Below is a list of terms we've defined in class. If you aren't comfortable with the meanings of each of these, you should probably look them up in the textbook or lecture slides.

- bias, variance, irreducible error
- unbiased (estimator)
- feature
- feature map
- linear least squares
- ridge
- lasso
- L1
- L2
- Linfinity
- regularization
- regularization path
- overfit

- underfit
- model/hypothesis class
- regression
- classification
- training set
- validation set
- test set
- cross-validation
- k-fold cross validation
- logistic regression
- sigmoid