446 Section 05

Plans for today!

1. This
2. Reminders
3. Subgradients
4. Midterm review

Reminders

- HW2 due May 6, 11:59 PM
  - Are you keeping track of late days? Use them!

- ***Midterm*** **on Friday May 1.**
  - Details on website and Ed…link
  - Good luck!

Question 1 and 2: K-fold CV + LASSO

Code on the website for questions 1 & 2 in the section 5 handout

- Check it out to see how k-fold cross validation is done in numpy
- Also a manual implementation of LASSO!

Subgradients

Why subgradients?

You can have convex functions that are not differentiable everywhere

- GD still useful to find minima
- But we may need to find subgradients

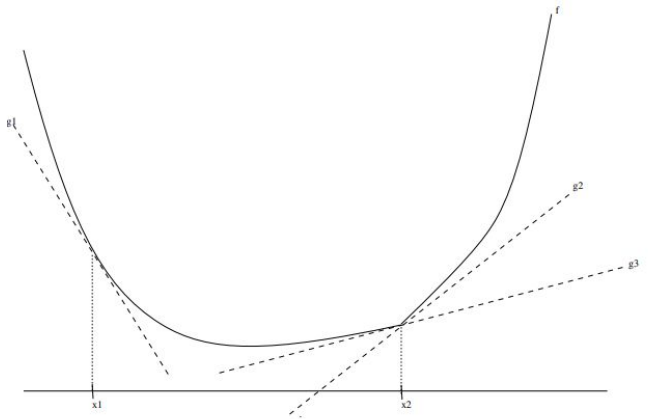
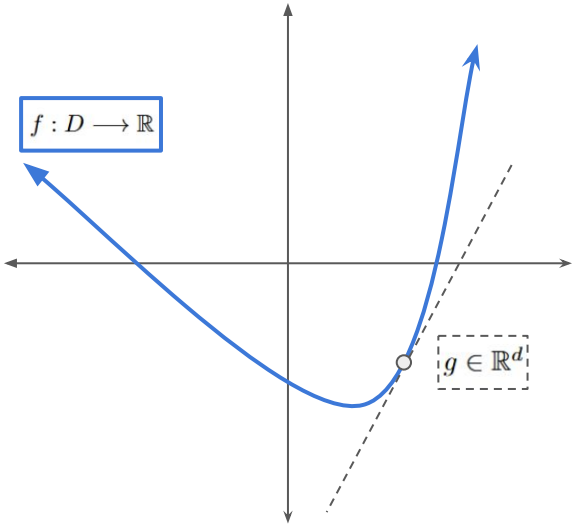


Figure 1: At x_1, the convex function f is differentiable, and g_1 (which is the derivative of f at x_1) is the unique subgradient at x_1. At the point x_2, f is not differentiable. At this point, f has many subgradients: two subgradients, g_2 and g_3, are shown.

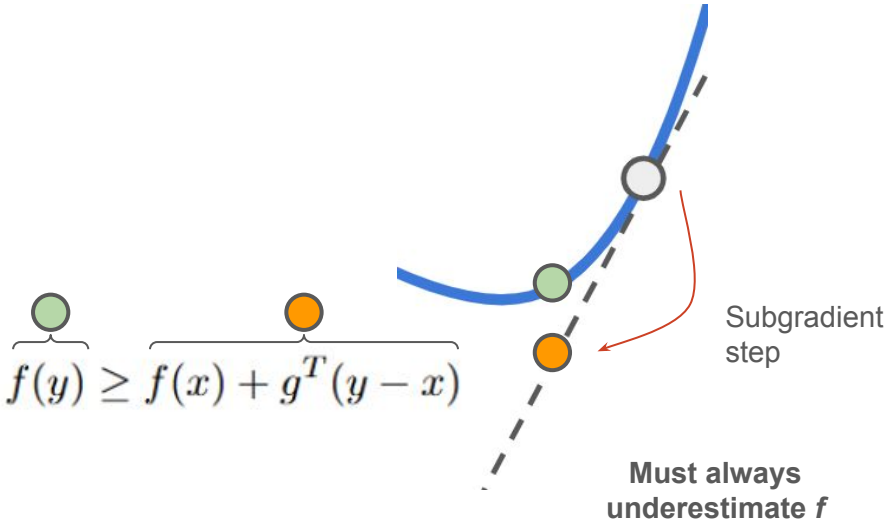
Subgradients visualized

Definition 1 (subgradients). *A vector $g \in \mathbb{R}^d$ is a subgradient of a convex function $f : D \longrightarrow \mathbb{R}$ at $x \in D \subseteq \mathbb{R}^d$ if*

$$f(y) \geq f(x) + g^T(y - x) \quad \textit{for all } y \in D.$$



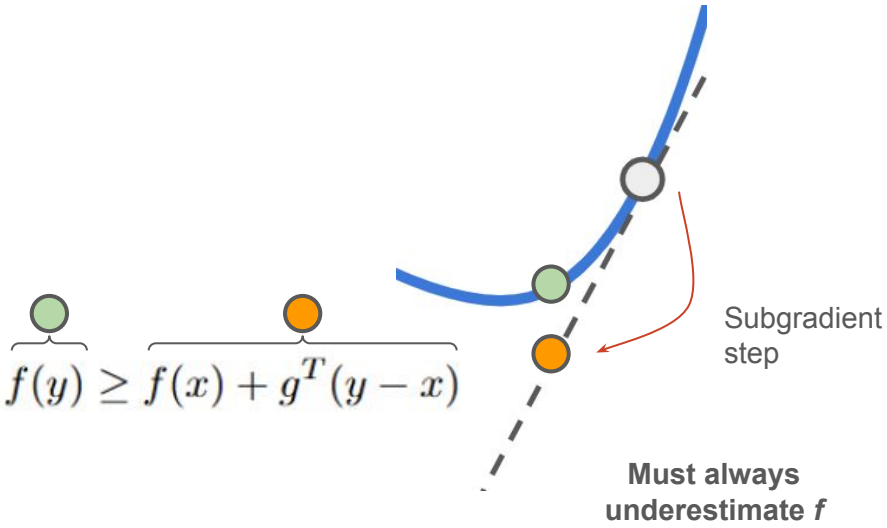
Zoom in



Subgradients visualized

Definition 1 (subgradients). *A vector $g \in \mathbb{R}^d$ is a subgradient of a convex function $f : D \longrightarrow \mathbb{R}$ at $x \in D \subseteq \mathbb{R}^d$ if*

$$f(y) \geq f(x) + g^T(y - x) \quad \textit{for all } y \in D.$$



Note:

- You can have many subgradients at a point *x*
- Subgradients can exist even at non-differentiable points *x*

Cool fact:

- If *f* **is** differentiable at *x*, then the gradient of *f* at *x* is also a subgradient of *f* at *x*

Question 3a

(a) Why are subgradients useful in optimization? If $g = 0$ is a subgradient of a function f at x^*, what does it imply?

Solution:

Consider the problem of minimizing a function f. If f is differentiable, we know that $\nabla f(x) = 0$ is a necessary condition for x to be a local extremum. Together with the convexity of f, $\nabla f(x) = 0$ becomes a sufficient condition for x to be a local minimizer, and hence global minimizer. If solving for $\nabla f(x) = 0$ analytically is difficult or infeasible, we have numerical methods such as gradient descent to obtain the solution(s). These results are very useful in minimizing a convex differentiable function and subgradients can be treated as a generalization to situations when the underlying function (to be minimized) is nondifferentiable. In the analytical aspect, if $g = 0$ is a subgradient of f at x^*, then from the definition above, we have

$$f(y) \geq f(x^*) + g^T(y - x^*) = f(x^*) \quad \text{for all } y \in D,$$

indicating that $f(x^*)$ is a global minimum. To obtain the solution(s) numerically, we can consider subgradient descent.

Question 3b

(b) What are the subgradients of $f(x) = \max(x, x^2)$ at 0, with $x \in \mathbb{R}$? (Hint: draw a picture and note that subgradients at a point might not be unique)

Solution:

From definition, a scalar g is a subgradient of f at $x = 0$ if $\max(y, y^2) = f(y) \geq f(0) + g(y - 0) = gy$ for any $y \in \mathbb{R}$. Solving $\max(y, y^2) \geq gy$ yields that $g \in [0, 1]$. That is, for any $g \in [0, 1]$, g is a subgradient of f at 0.

Midterm Review

Maximum Likelihood Estimation (MLE)

Observe $X_1, X_2, \ldots, X_n$ drawn IID from $f(x;\theta)$ for some "true" $\theta = \theta_*$

Likelihood function $L_n(\theta) = \prod_{i=1}^{n} f(X_i;\theta)$

Log-Likelihood function $l_n(\theta) = \log(L_n(\theta)) = \sum_{i=1}^{n} \log(f(X_i;\theta))$

Maximum Likelihood Estimator (MLE) $\widehat{\theta}_{MLE} = \arg\max_{\theta} L_n(\theta)$

Under benign assumptions, as the number of observations $n \to \infty$ we have $\widehat{\theta}_{MLE} \to \theta_*$

MLE - Practice

3. 10 points

The probability mass function of a geometric distribution with unknown parameter $0 < p \leq 1$ is

$$P(X = k|p) = (1-p)^{k-1}p,$$

where $k = 1, 2, 3, \ldots$. The interpretation of X is that it is the number of independent Bernoulli trials needed to get one success, if each trial has success probability p.

Given a set of n observations $\{x_1, x_2, \ldots, x_n\}$ from a geometric distribution, derive the Maximum Likelihood Estimate (MLE) $\hat{p}_{\text{MLE}}$ for the parameter p.

Hint: don't forget about the chain rule: for $h(x) = f(g(x))$, $h'(x) = f'(g(x))g'(x)$.

Answer: _____

MLE - Solution

$$\hat{p}_{MLE} = \operatorname*{argmax}_p \log P(x_1, \ldots, x_n \mid p)$$

$$= \operatorname*{argmax}_p \log \prod_{i=1}^{n} P(x_i \mid p)$$

$$= \operatorname*{argmax}_p \sum_{i=1}^{n} \log\left((1-p)^{x_i - 1} p\right)$$

$$= \operatorname*{argmax}_p \sum_{i=1}^{n} \left((x_i - 1)\log(1-p) + \log(p)\right)$$

MLE - Solution

$$\frac{d}{dp}\left(\sum_{i=1}^{n}\left((x_i-1)\log(1-p)+\log(p)\right)\right) = \sum_{i=1}^{n}\left(\frac{d}{dp}\left((x_i-1)\log(1-p)\right)+\frac{d}{dp}\log p\right)$$

$$= \sum_{i=1}^{n} \frac{1-x_i}{1-p} + \frac{1}{p}$$

$$= \sum_{i=1}^{n} \frac{p - px_i + 1 - p}{p - p^2}$$

$$= \sum_{i=1}^{n} \frac{1-px_i}{p(1-p)} = \sum_{i=1}^{n} \frac{1}{p(1-p)} - \frac{x_i}{1-p}$$

$$0 = \frac{n}{p(1-p)} - \frac{1}{1-p}\sum x_i$$

MLE - Solution

$$\frac{n}{p(1-p)} = \frac{1}{1-p}\sum x_i$$

$$\hat{p}_{MLE} = \frac{n}{\sum_{i=1}^{n} x_i}$$

Linear Regression (w/ Gaussian Noise)

Our first model!

- Utilized MLE to derive the SSE formula (we worked through this in lecture 3 and section 2):

$$\hat{w}_{MLE} = \arg\min_{w} \sum_{i=1}^{n} (y_i - x_i^\top w)^2$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix}$$

d : # of features
n : # of examples/datapoints

- Solve for w to get the closed form solution:

$$\begin{aligned} \hat{w}_{LS} &= \arg\min_{w} ||\mathbf{y} - \mathbf{X}w||_2^2 \\ &= \arg\min_{w} (\mathbf{y} - \mathbf{X}w)^T (\mathbf{y} - \mathbf{X}w) \end{aligned}$$

$$\hat{w}_{LS} = \hat{w}_{MLE} = \left(\boldsymbol{X}^T \boldsymbol{X}\right)^{-1} \boldsymbol{X}^T \boldsymbol{y}$$

Why LS? “Ordinary Least Squares”

Linear Regression (w/ Gaussian Noise) - Practice

9. 2 points | One Answer

In regression, when our prediction model is linear-Gaussian, i.e., $y_i \sim N(x_i^\top w, \sigma^2)$ for target output $y_i \in \mathbb{R}$ and feature vectors $x_i \in \mathbb{R}^d$, finding the w that maximizes the data likelihood is equivalent to minimizing the average absolute difference between the target output and predicted output.

a) True

b) False

Linear Regression (w/ Gaussian Noise) - Solution

Correct answers: (b)

Explanation: False because it would be minimizing the sum of squared differences, not absolute differences for linear-Gaussian.

Linear Regression w/ Basis Functions

- Linear models aren't able to properly fit non-linear data
- We can transform our features to a higher dimensional space and fit a linear model on it → Still linear regression, but on quadratic features

- **Data:** $\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$

- **Linear model with parameter** (b, w_1):
  - $\hat{y}_i = b + w_1 x_i$
- **Quadratic model with parameter** $(b, w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix})$
  - $\hat{y}_i = b + w_1 x_i + w_2 x_i^2$
- **Degree-p polynomial model with *p* parameters**
  - $\hat{y}_i = b + w_1 x_i + w_2 x_i^2 + \ldots + w_p x_i^p$

$$h(x_i) = \begin{bmatrix} 1 \\ x_i \\ x_i^2 \end{bmatrix}$$

$$\widehat{w} = \arg\min_w \|\mathbf{H}w - \mathbf{y}\|_2^2$$

For a new test point x, predict

$\hat{y} = \langle \widehat{w}, h(x) \rangle$

$$\hat{w}_{\text{MLE}} = (H^T H)^{-1} H^T y$$

Linear Regression w/ Basis Function - Practice

8. [2 points] [One Answer]

In a regression model, what is the primary purpose of using general basis functions?

(a) Transform nonlinear relationships between features and the target variable into a linear form.

(b) Regularize the model to prevent overfitting.

(c) Reduce the number of data samples needed for model training.

(d) Simplify the model by reducing the number of features.

Linear Regression w/ Basis Function - Solution

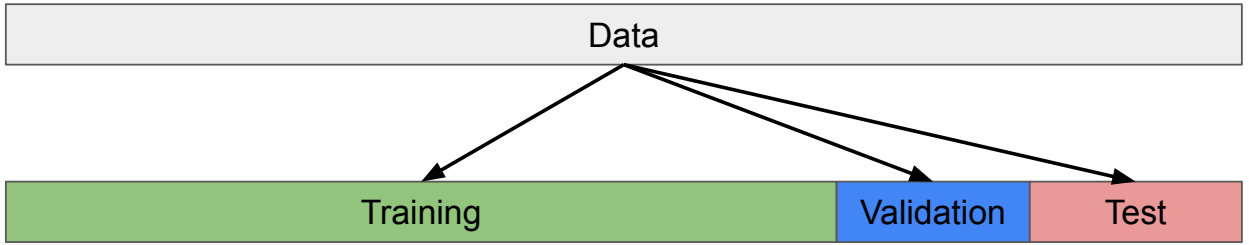
Correct answers: (a)

Explanation: (a) is correct because the primary purpose of using general basis functions in regression is to transform nonlinear relationships into a form that allows linear modeling techniques to be applied. By mapping features into a higher-dimensional space, basis functions can capture nonlinear patterns in the data. (b) is false because general basis functions alone do not perform regularization. (c) is false because using general basis functions typically does not reduce the number of samples required. In fact, using more complex basis functions often requires more data to fit the model accurately. (d) is false because general basis functions often increase the number of features by expanding the feature space (for example, by adding polynomials or interaction terms). This does not simplify the model; rather, it increases its complexity.

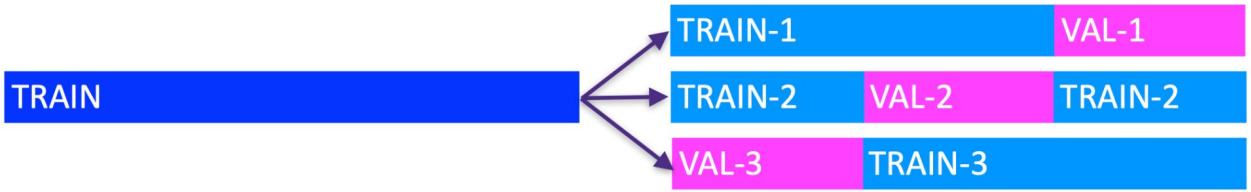
Train/val/test, cross validation

A predictor **generalizes** if it performs well on unseen data (and not just training data)

- Training set → used to learn general patterns
- Validation set → which of the trained models is the best?
- Test set → How good is this model really?



- Cross validation



Train/val/test, cross validation - Practice

A consortium of 10 hospitals have pooled together their Electronic Health Records data and want to build a machine learning model to predict patient prognosis based on patient records in their hospitals. They want to maximize the accuracy of their model across all 10 hospitals and do not plan to deploy their model in other hospitals. How should they split the data into train / validation / test sets?

(a) Leave out data from 1 hospital for the validation set, data from another hospital for the test set, and use the rest for train set.

(b) Leave out data from 1 hospital for the validation set, data from another hospital for the test set, and use the rest for train set. After training, add the validation data to the train set and re-train the model on the combined data.

(c) Use data from 8 hospitals with the most number of records for training, and use data from the other 2 hospitals for validation and test sets.

(d) Mix data from all hospitals, randomly shuffle all the records, and then do the 80/10/10 train/validation/test split.

Train/val/test, cross validation - Practice

Explanation: D is the correct answer, as it is the only approach that avoids overfitting hyperparameters to data from only one or two hospitals. Each hospital may have a different distribution of patients, doctors, outcomes, etc. So we should not expect all data to be IID.

Train/val/test, cross validation - Practice

In a project using a customer purchase history dataset with a 60/20/20 train, validation, and test split, the validation accuracy remains consistently lower than the training accuracy. What could be a reason for this?

Train/val/test, cross validation - Solution

Explanation: The validation accuracy is likely lower due to overfitting (the model is complex, variance is high). Overfitting happens when a model learns too much detail and noise from the training data, capturing specific patterns that don't apply to new, unseen data. This makes the model perform well on the training set but poorly on the validation or test sets, as it fails to generalize.

Bias variance trade-off

Theoretical optimal predictor

Ideally, we want to find:

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$$

But we only have samples:

$$(x_i, y_i) \overset{i.i.d.}{\sim} P_{XY} \quad \text{for } i = 1, \ldots, n$$

$$\widehat{f} = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2$$

Each draw $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ results in different $\widehat{f}$

Noisy estimate given limited samples n and possibly wrong model class f

$$\mathbb{E}_{Y|X}[\mathbb{E}_{\mathcal{D}}[(Y - \widehat{f}_{\mathcal{D}}(x))^2]|X = x]$$ ← *Generalization error*

$$= \mathbb{E}_{Y|X}[(Y - \eta(x))^2|X = x] + \mathbb{E}_{\mathcal{D}}[(\eta(x) - \widehat{f}_{\mathcal{D}}(x))^2]$$

irreducible error
Caused by stochastic label noise

learning error
Caused by either using too "simple" of a model or not enough data to learn the model accurately

Only thing in our control

$$= (\eta(x) - \mathbb{E}_{\mathcal{D}}[\widehat{f}_{\mathcal{D}}(x)])^2 + \mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{\mathcal{D}}[\widehat{f}_{\mathcal{D}}(x)] - \widehat{f}_{\mathcal{D}}(x))^2]$$

biased squared **variance**

Learning error = bias² + variance

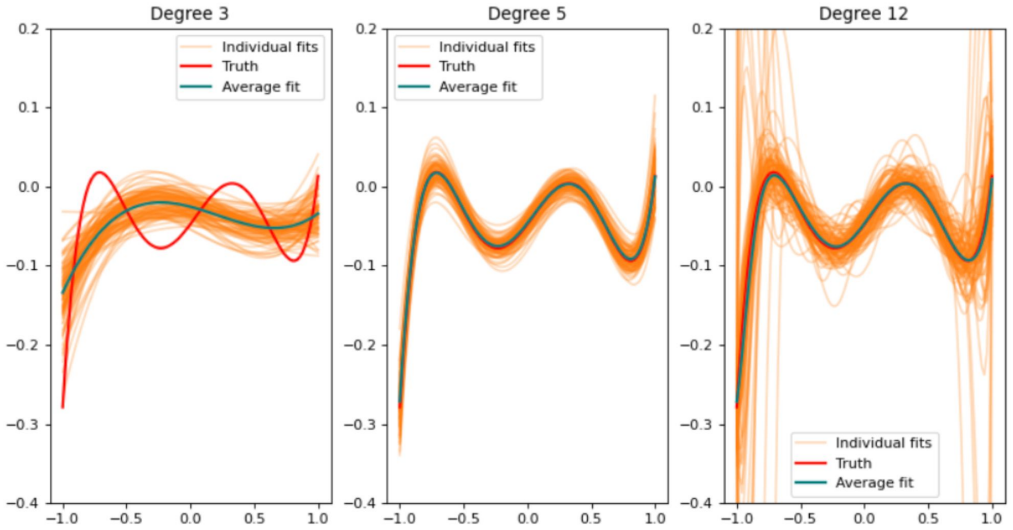
Bias variance trade-off

Bias: the difference between the true function and the average predictor

- High bias if the function class F can't describe data well → underfitting

Variance = for a particular dataset D ~ PXY, how diff is *f* from average *f ?*

- High variance if the difference is varying wildly → overfitting



High bias, low variance [underfit]

Low bias, low variance [good fit]

Low bias, high variance [overfit]

Bias variance trade-off - Practice

12. [2 points] [One Answer]

Which of the following best explains the effect of Lasso regression on the bias-variance tradeoff?

(a) Lasso regression reduces both bias and variance simultaneously, leading to a more accurate model.

(b) Lasso regression reduces bias by shrinking coefficients, often at the expense of increasing variance.

(c) Lasso regression reduces variance by shrinking coefficients and can increase bias, especially when some features are dropped entirely from the learned predictor.

(d) Lasso regression increases both bias and variance as it enforces sparsity in the learned predictor.

Bias variance trade-off - Solution

Correct answers: (c)

Explanation: The correct answer is (c) because Lasso regression penalizes the ℓ_1 norm of the weight vector, which shrinks coefficients (often to 0). This reduces the complexity of our model. A less complex model has higher bias and less variance. (a), (b), (d) are all incorrect because a less complex model has decreased variance.

Bias variance trade-off - Practice

1 points | One Answer

Suppose you train a linear regression model (without doing feature expansion), i.e., $f_w(x) = wx + b$, to approximate the cubic function $g(x) = 2x^3 + 7x^2 + 4x + 3$. What's the most likely outcome?

a) The model will have low bias and low variance

b) The model will have low bias and high variance

c) The model will have high bias and low variance

d) The model will have high bias and high variance

Bias variance trade-off - Solution

Correct answers: (c)

Explanation: The model being linear means the variance is likely to be low.
The linear model trying to approximate a cubic function, which is of a higher degree, means that the bias is likely to be high.

Bias variance trade-off - Practice

1 points | Select All That Apply

While training a model, you notice that it has a small bias but a high variance on the training data. Which of the following are valid strategies to address the high variance?

a. Increase regularization constant.

b. Train on a model class that is simpler.

c. Increase the size of the training dataset.

d. Use higher-degree features to capture more complex patterns in the data.

Bias variance trade-off - Solution

Correct answers: (a), (b), (c)

Explanation: A and B are correct because increasing regularization and simplifying model complexity help decrease the impact of less important features, improving generalization and reducing variance. C is also correct because increasing the training set size allows the model to generalize better, which can reduce variance. D is incorrect since using higher-degree features increases model complexity and often leads to overfitting, increasing variance.

Regularization: Ridge vs LASSO

Why do we use regularization?

- To address overfitting → generally increases bias and reduces variance – larger lambda = higher bias and lower variance (we formally proved this in lecture 7)

Ridge (L2 penalty):

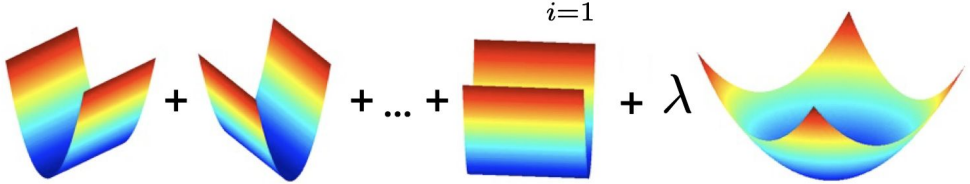
- Ensures invertibility for the closed form solution (even when d >> n, as we saw last week)
- Drives certain feature coefficients to low but non-zero values

LASSO (L1 penalty):

- Drives certain feature coefficients to zero
- Can be effectively used for feature selection → helps with model interpretability

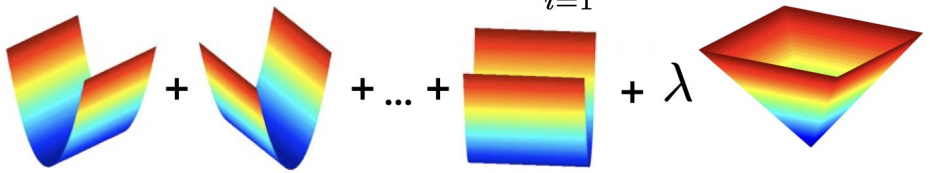
- Ridge Regression objective:

$$\widehat{w}_{ridge} = \arg\min_{w} \sum_{i=1}^{n} \left(y_i - x_i^T w\right)^2 + \lambda ||w||_2^2$$



- Lasso objective:

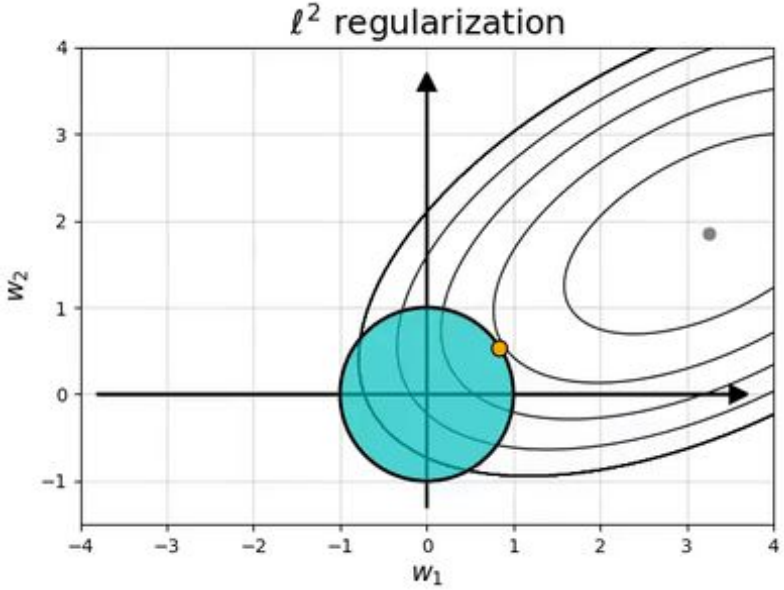
$$\widehat{w}_{lasso} = \arg\min_{w} \sum_{i=1}^{n} \left(y_i - x_i^T w\right)^2 + \lambda ||w||_1$$



Visualized: Ridge vs LASSO

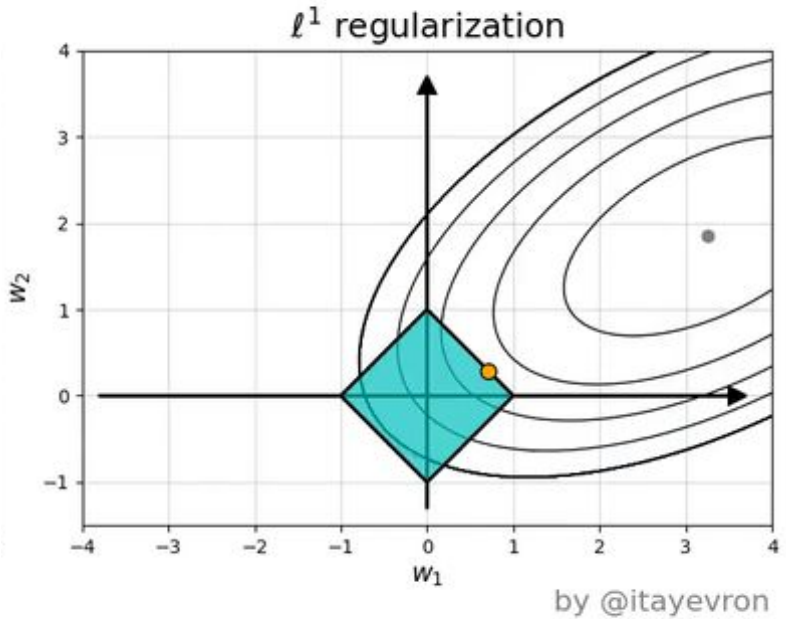
$$\widehat{w}_{ridge} = \arg\min_w \sum_{i=1}^{n} \left(y_i - x_i^T w\right)^2 + \lambda ||w||_2^2$$

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



$$\widehat{w}_{lasso} = \arg\min_w \sum_{i=1}^{n} \left(y_i - x_i^T w\right)^2 + \lambda ||w||_1$$

No closed form solution (see lecture 8 for why)!



Regularization: Ridge vs LASSO - Practice

11. [2 points] [One Answer]

You have a dataset with many features. You know *a priori* that only a small portion of those features are relevant to your prediction problem, but you don't know which are the relevant features. Is it better to use Ridge regression or Lasso regression?

(a) Ridge regression

(b) Lasso regression

Regularization: Ridge vs LASSO - Solution

Correct answers: (b)

Explanation: The correct answer is (b), Lasso regression, because Lasso uses L1 regularization while Ridge uses L2. L1 penalizes all weights at the same rate unlike L2, so it encourages higher sparsity in the weights. We want higher sparsity in the weights because we know beforehand that only a small portion of the features are actually relevant. So, we want only a small portion of features to have weight that is not 0. If we used L2 regularization, then more features would have non-zero weight and we would assign meaning to many features that should not have any based on our a priori knowledge.

Regularization: Ridge vs LASSO - Practice

10. [6 points] [Select All That Apply]

In ridge regression, we obtain $\widehat{w}_{\text{ridge}} = (X^T X + \lambda I)^{-1} X^T y$ for $\lambda \geq 0$. Which of the following is **true**?

(a) $X^T X$ is always invertible.

(b) $X^T X + \lambda I$ is always invertible.

(c) Increasing λ typically adds bias to the model.

(d) Increasing λ typically adds variance to the model.

(e) When $\lambda = 0$, ridge regression reduces to ordinary (unregularized) linear regression.

(f) As $\lambda \to \infty$, $\widehat{w}_{\text{ridge}} \to 0$.

Regularization: Ridge vs LASSO - Solution

Correct answers: (c), (e), (f)

Explanation: (a) is incorrect because $X^T X$ is positive semi-definite, which is not always invertible if X has a non-empty null space. (b) is incorrect because when λ is 0, it can be reduced to (a). (c) is correct since increasing λ results in an increase to $\lambda^2/(n+\lambda)^2(w^T x)^2$ in the biased square term when n is large. Conceptually, the model penalizes large weights, pulling them closer to zero. This constraint often reduces the model's flexibility, which adds bias. (d) is incorrect since increasing λ results in a decrease to $\sigma^2 n/(n+\lambda)^2\|x\|_2^2$. Conceptually, it makes the model it less sensitive to fluctuations in the training data, which lowers variance at the cost of potentially increasing bias. (e) is correct by definition of OLS. (f) is correct because the regularization term dominates, causing the ridge regression to shrink toward zero.

Regularization: Ridge vs LASSO - Practice

1 points | One Answer

Which of the following statements about ridge regression are true?

a When there are correlated features, ridge regression typically sets the weights of all but one of the correlated features to 0.

b Compared to unregularized linear regression, the additional computational cost of ridge regression increases proportional to the number of data points in the dataset.

c Ridge regression reduces variance at the expense of increasing bias.

d Using ridge and lasso regularization together (e.g., minimizing a training objective of the form $f(w) = \sum_{i=1}^{n}(y^{(i)} - x^{(i)\top}w)^2 + \lambda_1\|w\|_1 + \lambda_2\|w\|_2^2$) makes the training loss no longer convex.

Regularization: Ridge vs LASSO - Solution

Correct answers: (c)

Explanation: Ridge regression typically shrinks the weights of correlated features about evenly. This means it probably won't set the weights of all but one of the correlated features to 0. That would be more akin to LASSO regression. The additional computational cost increases proportional to the number of weights in the dataset. Ridge-regression is an example of the bias-variance trade off. The sum of convex functions is convex so ridge and LASSO regression combined is still convex.

Convexity

Why is convexity important?

- Any local minimum is a global minimum → enables reliable optimization

 # Sets

A set $K \subset \mathbb{R}^d$ is convex if $(1-\lambda)x + \lambda y \in K$ for all $x, y \in K$ and $\lambda \in [0,1]$

 # Line above function

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if $f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$ for all $x, y \in \mathbb{R}^d$ and $\lambda \in [0,1]$

 # Epigraph

A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if the set $\{(x,t) \in \mathbb{R}^{d+1} : f(x) \leq t\}$ is convex

 # 1st order Taylor expansion below function

A function $f : \mathbb{R}^d \to \mathbb{R}$ that is differentiable everywhere is convex if $f(y) \geq f(x) + \nabla f(x)^\top (y - x)$ for all $x, y \in dom(f)$

 # Hessian

A function $f : \mathbb{R}^d \to \mathbb{R}$ that is twice-differentiable everywhere is convex if $\nabla^2 f(x) \succeq 0$ for all $x \in dom(f)$

Convexity - Practice

23. 4 points | Select All That Apply

Which of the following are **true** about a convex function $f(x) : \mathbb{R}^d \to \mathbb{R}$?

a $f(x)$ must be differentiable across its entire domain.

b $f(x)$ has a unique global minimum.

c $g(x) = -f(x)$ is also convex.

d If $f(x)$ is twice differentiable, then $z^\top \nabla^2 f(x) z \geq 0$ for all $z \in \mathbb{R}^d$.

Convexity - Solution

Correct answers: (d)

Explanation: The correct answer(s) are: D. A is incorrect–consider the function f(x) = |x|. This is convex but is not differentiable at x=0. B is incorrect because a convex function may have multiple connected global minima (e.g., the "half-pipes" we discussed when building up ridge regression) or no global minima (e.g., a hyperplane with non-zero slope). C is only true when f(x) is a linear or affine function, but is not true in general (e.g., a bowl is convex, but when you flip it upside down it becomes concave).

Convexity - Practice

24. 5 points | Select All That Apply

Which of the following have convex objective functions?

a Linear regression

b Linear regression with arbitrary nonlinear basis functions

c Ridge regression

d Lasso regression

e Logistic regression

Convexity - Solution

Correct answers: (a), (b), (c), (d), (e)

Explanation: All the aforementioned models use a linear function to map inputs to outputs, and their objective function is linear.

Gradient Descent

- Iterative optimization method that can be used when we don't have a closed form solution.
- Any local minima we find are guaranteed to be globally optimal iff function is convex.
- Worked through a theoretical analysis of GD in lecture 11

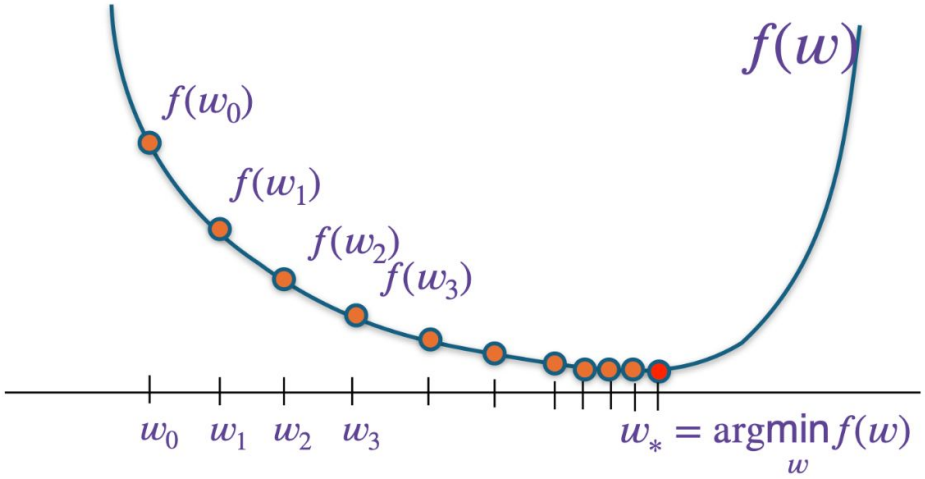
Algorithm

For t=0,1,2,3, …

$$w_{t+1} = w_t - \eta \frac{df(w)}{dw}|_{w=w_t}$$

Hyperparameters:

- Initial point w_0
- Step size η



Note that as $t \to \infty$ we have $\frac{df(w)}{dw}|_{w=w_t} \to 0$

Gradient Descent - Practice

21. [4 points] [Select All That Apply]

Given the gradient descent algorithm, $w_{t+1} = w_t - \eta \frac{df(w)}{dw}\Big|_{w=w_t}$, which of the following statement is correct regarding the hyperparameter η?

(a) η controls the magnitude of each step.

(b) η determines the initial value of w.

(c) A larger η guarantees faster convergence to the global minimum.

(d) A smaller η guarantees faster convergence to the global minimum.

Gradient Descent - Solution

Correct answers: (a)

Explanation: η controls the step size in the gradient descent algorithm. η and w are independently set. A larger η may cause model update to overshoot the global minimum. A smaller η may cause model to get stuck in local minimum.

SGD and Minibatch SGD

Gradient descent: $w_{t+1} = w_t - \eta \nabla_w \left(\frac{1}{n} \sum_{i=1}^{n} \ell_i(w) \right) \Big|_{w=w_t}$ # Entire training dataset

Stochastic gradient descent: $w_{t+1} = w_t - \eta \nabla_w \ell_{I_t}(w) \big|_{w=w_t}$

n times faster per iteration I_t drawn uniformly at random from $\{1, \ldots, n\}$

Instead of one iterate, average B stochastic gradients together

Sample a batch of data $D_B = \{x_i \sim D : i = 1, \ldots, B\}, \quad B \ll N$

Advantages:

- Smaller variance (by $1/B$)
- Parallelization: Each gradient in the minibatch can be computed in parallel

SGD and Minibatch SGD - Practice

20. [2 points] [One Answer]

For a possibly non-convex optimization problem, gradient descent on the full dataset always finds a better solution than stochastic gradient descent.

(a) True

(b) False

SGD and Minibatch SGD - Solution

Correct answers: (b)

Explanation: Gradient descent is **not** always better than stochastic gradient descent. The variability of SGD can escape local minima more effectively than deterministic gradient descent.

Prediction Pitfalls

1) Misinterpreting coefficients: bigger weights don't mean more important features. Zero weight doesn't mean no signal. Coefficients reflect correlation, not causation.

2) Distribution shift (OOD): test accuracy on held-out data is not a guarantee of real-world performance. New hospitals, geographies, time periods → the model breaks.

3) Biases in the data: reporting bias, proxy labels (utilization != risk), and historical human decisions baked into training data all corrupt models → even with good intentions (Boston potholes, Amazon hiring).

4) Removing features isn’t enough: models recover certain attributes via correlated proxies (e.g., zip code). Removing race/gender can sometimes hurt minority groups → thoughtful feature engineering is better.

Prediction Pitfalls - Practice

19. [2 points] [One Answer]

You are building a predictive model about users of a website. Suppose that after you train your model on historical user data, the distribution of users shifts dramatically. What can happen if you deploy your machine learning system without addressing this distribution shift?

(a) The model will automatically adapt to new data distributions.

(b) The model will generate more diverse predictions, increasing its overall accuracy.

(c) The model will maintain its original performance indefinitely regardless of data changes.

(d) The model's predictions may become unreliable or biased.

Prediction Pitfalls - Solution

Correct answers: (d)

Explanation: Machine learning models can only reliably generalize to data from the same distribution they were trained on; when faced with different distributions, their predictions may become unreliable or biased due to this domain shift, rather than becoming more diverse or accurate.

Prediction Pitfalls - Practice

18. [2 points]

Given the task of determining loan approval for applicants using a predictive model given applicant features such as race, salary, education, etc., is it always best practice to allow the model to use all of the given features? Why or why not?

Prediction Pitfalls - Solution

Explanation: No, we should not ALWAYS use all the features. In addition to building an accurate model, we also want to build ethically-informed models and this requires us to be thoughtful about what features go into our analyses. For any feature we choose to include, our model may find correlations that are not necessarily causations, that are either coincidental or the result of pre-existing biases. Depending on the most informed choice to make, the best practice may or may not be to include all available features.

Solutions

Additional Midterm Review!

Questions/Chat Time!