# Homework #2

CSE 446/546: Machine Learning
Professors Pang Wei Koh & Sewoong Oh
Due: **Wednesday** November 5, 2025 11:59pm
**A**: 106 points, **B**: 20 points

**Jiayao Huang**

Please review all homework guidance posted on the website before submitting it to Gradescope. Reminders:

- All code must be written in Python and all written work must be typeset (e.g. LaTeX).
- Make sure to read the "What to Submit" section following each question and include all items.
- Please provide succinct answers and supporting reasoning for each question. Similarly, when discussing experimental results, concisely create tables and/or figures when appropriate to organize the experimental results. All explanations, tables, and figures for any particular part of a question must be grouped together.
- For every problem involving generating plots, please include the plots as part of your PDF submission.
- When submitting to Gradescope, please link each question from the homework in Gradescope to the location of its answer in your homework PDF. Failure to do so may result in deductions of up to 10% of the value of each question not properly linked. For instructions, see `https://www.gradescope.com/get_started#student-submission`.

Not adhering to these reminders may result in point deductions.

**Important:** By turning in this assignment (and all that follow), you acknowledge that you have read and understood the collaboration policy with humans and AI assistants alike: `https://courses.cs.washington.edu/courses/cse446/25au/assignments/`. Any questions about the policy should be raised at least 24 hours before the assignment is due. There are no warnings or second chances. If we suspect you have violated the collaboration policy, we will report it to the college of engineering who will complete an investigation.
Not adhering to these reminders may result in point deductions.

## Conceptual Questions

A1. The answers to these questions should be answerable without referring to external materials. Briefly justify your answers with a few words.

a. *[2 points]* Suppose that your estimated model for predicting house prices has a large positive weight on the feature `number of bathrooms`. If we remove this feature and refit the model, will the new model have a strictly higher error than before? Why?

b. *[2 points]* Compared to L2 norm penalty, explain why a L1 norm penalty is more likely to result in sparsity (a larger number of 0s) in the weight vector.

c. *[2 points]* In at most one sentence each, state one possible upside and one possible downside of using the following regularizer: $\left(\sum_i |w_i|^{0.5}\right)$.

d. *[1 point]* True or False: If the step-size for gradient descent is too large, it may not converge.

e. *[2 points]* In your own words, describe why stochastic gradient descent (SGD) works, even though only a small portion of the data is considered at each update.

f. *[2 points]* In at most one sentence each, state one possible advantage of SGD over GD (gradient descent), and one possible disadvantage of SGD relative to GD.

### What to Submit:

- **Part d:** True or False.
- **Parts a-f:** Brief (2-3 sentence) explanation.

`Part a.` No, not necessarily. The error could remain the same or even decrease if the removed feature is redundant and its predictive information is captured by other correlated features in the model.

`Part b.` The geometry of the L1 constraint makes it more likely that the optimum lies at a point where some coefficients are zero, creating sparsity. In contrast, the L2 penalty usually shrinks all weights uniformly but seldom drives them to exactly zero.

`Part c.` Upside: It can promote even stronger sparsity than the L1 norm, potentially driving more weights exactly to zero.
Downside: The function is not convex, making optimization more difficult and prone to local minima.

`Part d.` True.

`Part e.` SGD works because the noisy gradient computed from a single data point is an unbiased estimate of the true gradient. On average, over many updates, it progresses in the correct direction towards the minimum.

`Part f.` Advantage: SGD is much faster in each iteration and can optimize more quickly, especially on large datasets.
Disadvantage: SGD has high variance in its parameter updates, causing the convergence path to be noisy and may prevent it from settling at the global minimum.

# Convexity and Norms

A2. A *norm* $\|\cdot\|$ over $\mathbb{R}^n$ is defined by the properties: $(i)$ non-negativity: $\|x\| \geq 0$ for all $x \in \mathbb{R}^n$ with equality if and only if $x = 0$, $(ii)$ absolute scalability: $\|a\,x\| = |a|\,\|x\|$ for all $a \in \mathbb{R}$ and $x \in \mathbb{R}^n$, $(iii)$ triangle inequality: $\|x + y\| \leq \|x\| + \|y\|$ for all $x, y \in \mathbb{R}^n$.

a. *[3 points]* Show that $f(x) = (\sum_{i=1}^n |x_i|)$ is a norm. (Hint: for $(iii)$, begin by showing that $|a+b| \leq |a|+|b|$ for all $a, b \in \mathbb{R}$.)

b. *[2 points]* Show that $g(x) = \left(\sum_{i=1}^n |x_i|^{1/2}\right)^2$ is not a norm. (Hint: it suffices to find two points in $n = 2$ dimensions such that the triangle inequality does not hold.)

Context: norms are often used in regularization to encourage specific behaviors of solutions. If we define $\|x\|_p := (\sum_{i=1}^n |x_i|^p)^{1/p}$ then one can show that $\|x\|_p$ is a norm for all $p \geq 1$. The important cases of $p = 2$ and $p = 1$ correspond to the penalty for ridge regression and the lasso, respectively.

## What to Submit:

- **Parts a, b:** Proof.

`Proof.` (a)
(i) Non-negativity: Since $|x_i| \geq 0$ for all $i$, we have $f(x) = \sum_{i=1}^n |x_i| \geq 0$.
If $f(x) = 0$, then

$$\sum_{i=1}^n |x_i| = 0 \quad \Rightarrow \quad |x_i| = 0, \text{ for all } i \quad \Rightarrow \quad x_i = 0, \text{ for all } i \quad \Rightarrow \quad x = 0$$

Conversely, if $x = 0$, then $f(x) = 0$.
(ii) Absolute scalability: For any $a \in \mathbb{R}$, $x \in \mathbb{R}^n$,

$$f(ax) = \sum_{i=1}^n |ax_i| = \sum_{i=1}^n |a||x_i| = |a| \sum_{i=1}^n |x_i| = |a| f(x).$$

(iii) Triangle inequality: We first show that, for any $a, b \in \mathbb{R}$, $|a + b| \leq |a| + |b|$.
On one hand,

$$|a + b|^2 = a^2 + 2ab + b^2$$

On the other hand,

$$(|a| + |b|)^2 = a^2 + 2|a||b| + b^2$$

Note that $2ab \leq 2|a||b|$, we have

$$|a + b|^2 \leq (|a| + |b|)^2$$

Since $|a + b| \geq 0$, $|a| + |b| \geq 0$, we get $|a + b| \leq |a| + |b|$.
Then for $x, y \in \mathbb{R}^n$:

$$f(x + y) = \sum_{i=1}^n |x_i + y_i| \leq \sum_{i=1}^n (|x_i| + |y_i|) = \sum_{i=1}^n |x_i| + \sum_{i=1}^n |y_i| = f(x) + f(y).$$

Thus, by (i), (ii), (iii), $f(x)$ is a norm.

**(b)** Consider $g(x) = \left(\sum_{i=1}^{2} |x_i|^{1/2}\right)^2$ in $\mathbb{R}^2$. If we take $x = (1, 0)$ and $y = (0, 1)$, then

$$g(x) = (|1|^{1/2} + |0|^{1/2})^2 = (1 + 0)^2 = 1$$

$$g(y) = (|0|^{1/2} + |1|^{1/2})^2 = (0 + 1)^2 = 1$$

Now

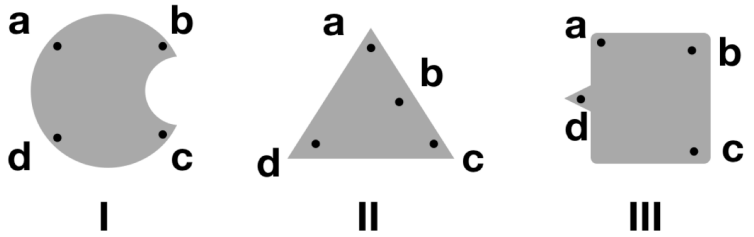
$$g(x) + g(y) = 2$$

But

$$g(x + y) = g(1, 1) = (|1|^{1/2} + |1|^{1/2})^2 = (1 + 1)^2 = 4$$

The triangle inequality requires $g(x + y) \leq g(x) + g(y)$, but now we have $4 \leq 2$, which is not true. Thus $g(x) = \left(\sum_{i=1}^{n} |x_i|^{1/2}\right)^2$ is not a norm.

A3. *[3 points]* A set $A \subseteq \mathbb{R}^n$ is *convex* if $\lambda x + (1-\lambda)y \in A$ for all $x, y \in A$ and $\lambda \in [0,1]$. For each of the grey-shaded sets below (I-III), state whether each one is convex, or state why it is not convex using any of the points $a, b, c, d$ in your answer.




## What to Submit:

- **Parts I-III:** 1-2 sentence explanation of why the diagram is convex or not.

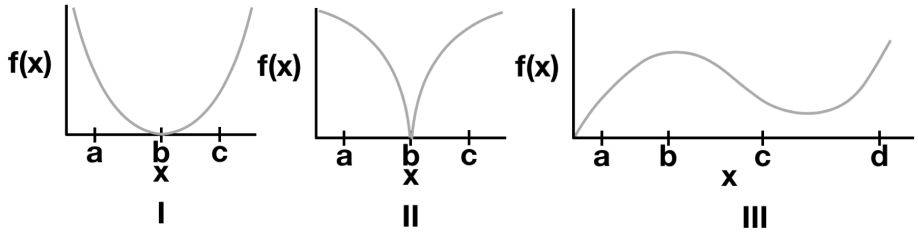
`Solve.`

**I. Not convex.**
If we link points $b$ and $c$, the straight line segment crosses the the white region, so the segment is not contained in the shaded set.

**II. Convex.**
This shaded region is a triangle. For any two pointsin the diagram, every convex combination lies inside the triangle, so every segment between two points of the set stays inside the set.

**III. Not convex.**
If we link points $a$ and $d$, the straight segment between them passes through the white region, so it is not contained in the shaded region.

A4. *[4 points]* We say a function $f : \mathbb{R}^d \to \mathbb{R}$ is convex on a set $A$ if $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$ for all $x, y \in A$ and $\lambda \in [0, 1]$. For each of the functions shown below (I-III), state whether each is convex on the specified interval, or state why not with a counterexample using any of the points $a, b, c, d$ in your answer.




a. Function in panel I on $[a, c]$

b. Function in panel II on $[a, c]$

c. Function in panel III on $[a, d]$

d. Function in panel III on $[c, d]$

## What to Submit:

- **Parts a-d:** 1-2 sentence explanation of why the function is convex or not.

`Solve.`
**Part a:** Convex.
The graph is U-shaped with its minimum at $b$, so every straight line between two points (e.g. $a$ and $c$) lies above the curve. In particular $f(b) < \frac{f(a)+f(c)}{2}$, so the convexity inequality holds.

**Part b:** Not convex.
Taking $x = a$, $y = b$ and $\lambda = \frac{1}{2}$, the function value at the midpoint is above the line segment joining $(a, f(a))$ and $(b, f(b))$, i.e.

$$f\big(\frac{a+b}{2}\big) > \frac{f(a)+f(b)}{2}$$

If we take $x = b$, $y = c$ and $\lambda = \frac{1}{2}$, we have the same result.
Thus the function is not convex.

**Part c:** Not convex.
Taking $x = a$, $y = c$ and $\lambda = \frac{1}{2}$, the function value at the midpoint is above the line segment joining $(a, f(a))$ and $(b, f(b))$, i.e.

$$f\big(\frac{a+c}{2}\big) > \frac{f(a)+f(c)}{2}$$

Thus the function is not convex.

**Part d:** Convex.
On interval $[c, d]$, the graph is U-shaped, so every straight line between two points (e.g. $f(c)$ and $f(d)$) lies above the curve. In particular $f\big(\frac{c+d}{2}\big) < \frac{f(c)+f(d)}{2}$, so the convexity inequality holds.

B1. Use just the definitions above and let $\|\cdot\|$ be a norm.

a. *[3 points]* Show that $f(x) = \|x\|$ is a convex function.

b. *[3 points]* Show that $\{x \in \mathbb{R}^n : \|x\| \leq 1\}$ is a convex set.

c. *[2 points]* Draw a picture of the set $\{(x_1, x_2) \ : \ g(x_1, x_2) \leq 4\}$ where $g(x_1, x_2) = \left(|x_1|^{1/2} + |x_2|^{1/2}\right)^2$. (This is the function considered in 1b above specialized to $n = 2$.) We know $g$ is not a norm. Is the defined set convex? Why not?

Context: It is a fact that a function $f$ defined over a set $A \subseteq \mathbb{R}^n$ is convex if and only if the set $\{(x, z) \in \mathbb{R}^{n+1} : z \geq f(x), x \in A\}$ is convex. Draw a picture of this for yourself to be sure you understand it.

## What to Submit:

- **Parts a, b:** Proof.
- **Part c:** A picture of the set, and 1-2 sentence explanation.

`Proof.` By definition, we need to show that for all $x, y \in \mathbb{R}^n$ and $\lambda \in [0, 1]$:

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y).$$

Using the triangle inequality and absolute scalability properties of the norm, we have

$$\begin{aligned} f(\lambda x + (1-\lambda)y) &= \|\lambda x + (1-\lambda)y\| \\ &\leq \|\lambda x\| + \|(1-\lambda)y\| \quad \text{(triangle inequality)} \\ &= |\lambda|\|x\| + |1-\lambda|\|y\| \end{aligned}$$

Since $\lambda \in [0, 1]$, $1 - \lambda \in [0, 1]$, then

$$|\lambda|\|x\| + |1-\lambda|\|y\| = \lambda\|x\| + (1-\lambda)\|y\| = \lambda f(x) + (1-\lambda) f(y)$$

Thus,

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$$

Therefore, by definition, $f(x) = \|x\|$ is convex.

`Proof.` Let $C = \{x \in \mathbb{R}^n : \|x\| \leq 1\}$. For any $x, y \in C$ and $\lambda \in [0, 1]$. We need to show $\lambda x + (1-\lambda)y \in C$.

$$\begin{aligned} \|\lambda x + (1-\lambda)y\| &\leq \|\lambda x\| + \|(1-\lambda)y\| \quad \text{(triangle inequality)} \\ &= |\lambda|\|x\| + |1-\lambda|\|y\| \\ &= \lambda\|x\| + (1-\lambda)\|y\| \end{aligned}$$

Since $x, y \in C$, we further have

$$\lambda\|x\| + (1-\lambda)\|y\| \leq \lambda \cdot 1 + (1-\lambda) \cdot 1 = 1.$$

Thus

$$\|\lambda x + (1-\lambda)y\| \leq 1 \quad \Rightarrow \quad \lambda x + (1-\lambda)y \in C$$

Therefore, $C = \{x \in \mathbb{R}^n : \|x\| \leq 1\}$ is convex.

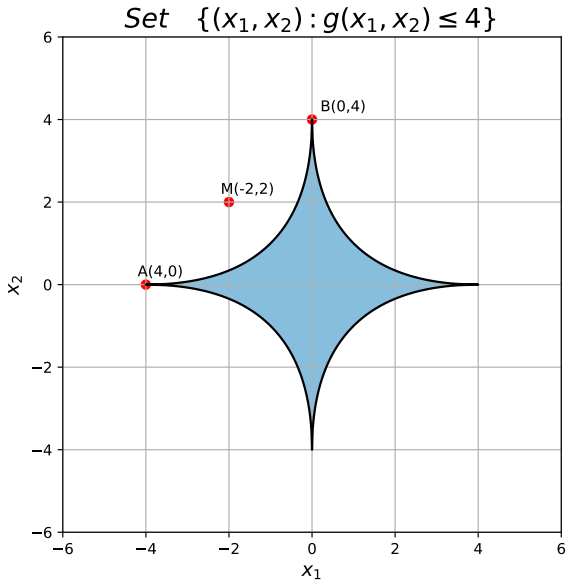



Figure 1: Picture of the set $\{(x_1, x_2) \ : \ g(x_1, x_2) \leq 4\}$

`Solve.` The picture is shown in Figure 1.
The set is not convex. If we connect two points $A = (-4, 0)$ and $B = (0, 4)$ in this set with a straight line, the midpoint $M$ gives $g(-2, -2) = (\sqrt{2} + \sqrt{2})^2 = 8 > 4$, so the midpoint lies outside the set.
Thus by definition, the set is not convex.

## Lasso on a Real Dataset

Given $\lambda > 0$ and data $\left(x_i, y_i\right)_{i=1}^n$, the Lasso is the problem of solving

$$\arg \min_{w \in \mathbb{R}^d, b \in \mathbb{R}} \sum_{i=1}^n (x_i^T w + b - y_i)^2 + \lambda \sum_{j=1}^d |w_j|$$

where $\lambda$ is a regularization parameter. For the programming part of this homework, you will implement the iterative shrinkage thresholding algorithm shown in Algorithm 1 to solve the Lasso problem in `ISTA.py`. This is a variant of the subgradient descent method and a more detailed discussion can be found in these slides. You may use common computing packages (such as `numpy` or `scipy`), but do not use an existing Lasso solver (e.g., of `scikit-learn`).
You may use common computing packages (such as `numpy` or `scipy`), but do not use an existing Lasso solver (e.g., of `scikit-learn`).

**Algorithm 1:** Iterative Shrinkage Thresholding Algorithm for Lasso

**Input:** Step size $\eta$
**while** *not converged* **do**
  $b' \leftarrow b - 2\eta \sum_{i=1}^n (x_i^T w + b - y_i)$
  **for** $k \in \{1, 2, \cdots d\}$ **do**
    $w_k' \leftarrow w_k - 2\eta \sum_{i=1}^n x_{i,k}(x_i^T w + b - y_i)$
    $$w_k' \leftarrow \begin{cases} w_k' + 2\eta\lambda & w_k' < -2\eta\lambda \\ 0 & w_k' \in [-2\eta\lambda, 2\eta\lambda] \\ w_k' - 2\eta\lambda & w_k' > 2\eta\lambda \end{cases}$$
  **end**
  $b \leftarrow b'$, $w \leftarrow w'$
**end**

Before you get started, the following hints may be useful:

- Wherever possible, use matrix libraries for matrix operations (not `for` loops).
- There are opportunities to considerably speed up parts of the algorithm by precomputing quantities like $a_k$ before the for loop; you are permitted to add these improvements (and it may save you some time).
- As a sanity check, ensure the objective value is nonincreasing with each step.
- It is up to you to decide on a suitable stopping condition. A common criteria is to stop when no element of $w$ changes by more than some small $\delta$ during an iteration. If you need your algorithm to run faster, an easy place to start is to loosen this condition.
- You will need to solve the Lasso on the same dataset for many values of $\lambda$. This is called a regularization path. One way to do this efficiently is to start at a large $\lambda$, and then for each consecutive solution, initialize the algorithm with the previous solution, decreasing $\lambda$ by a constant ratio (e.g., by a factor of 2).
- The smallest value of $\lambda$ for which the solution $\widehat{w}$ is entirely zero is given by

$$\lambda_{max} = \max_{k=1,\ldots,d} 2 \left| \sum_{i=1}^n x_{i,k} \left( y_i - \left( \frac{1}{n} \sum_{j=1}^n y_j \right) \right) \right| \tag{1}$$

  This is helpful for choosing the first $\lambda$ in a regularization path.

A5. We will first try out your solver with some synthetic data. A benefit of the Lasso is that if we believe many features are irrelevant for predicting $y$, the Lasso can be used to enforce a sparse solution, effectively

differentiating between the relevant and irrelevant features. Suppose that $x \in \mathbb{R}^d, y \in \mathbb{R}, k < d$, and data are generated independently according to the model $y_i = w^T x_i + \epsilon_i$ where

$$w_j = \begin{cases} j/k & \text{if } j \in \{1, \ldots, k\} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

and $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ is noise (note that in the model above $b = 0$). We can see from Equation (2) that since $k < d$ and $w_j = 0$ for $j > k$, the features $k+1$ through $d$ are irrelevant for predicting $y$.
Generate a dataset using this model with $n = 500, d = 1000, k = 100$, and $\sigma = 1$. You should generate the dataset such that each $\epsilon_i \sim \mathcal{N}(0, 1)$ , and $y_i$ is generated as specified above. You are free to choose a distribution from which the $x$'s are drawn, but make sure standardize the $x$'s before running your experiments.

a. *[10 points]* With your synthetic data, solve multiple Lasso problems on a regularization path, starting at $\lambda_{max}$ where no features are selected (see Equation (1)) and decreasing $\lambda$ by a constant ratio (e.g., 2) until nearly all the features are chosen. In plot 1, plot the number of non-zeros as a function of $\lambda$ on the x-axis (Tip: use `plt.xscale('log')`).

b. *[10 points]* For each value of $\lambda$ tried, record values for false discovery rate (FDR) (number of incorrect nonzeros in $\widehat{w}$/total number of nonzeros in $\widehat{w}$) and true positive rate (TPR) (number of correct nonzeros in $\widehat{w}$/k). Note: for each $j$, $\widehat{w}_j$ is an incorrect nonzero if and only if $\widehat{w}_j \neq 0$ while $w_j = 0$. In plot 2, plot these values with the x-axis as FDR, and the y-axis as TPR.

   Note that in an ideal situation we would have an (FDR,TPR) pair in the upper left corner. We can always trivially achieve $(0, 0)$ and $(\frac{d-k}{d}, 1)$.

c. *[5 points]* Comment on the effect of $\lambda$ in these two plots in 1-2 sentences.

## What to Submit:

- **Part a:** Plot 1.
- **Part b:** Plot 2.
- **Part c:** 1-2 sentence explanation.
- **Code** on Gradescope through coding submission

`Part a.` The plot is displayed in Figure 2.

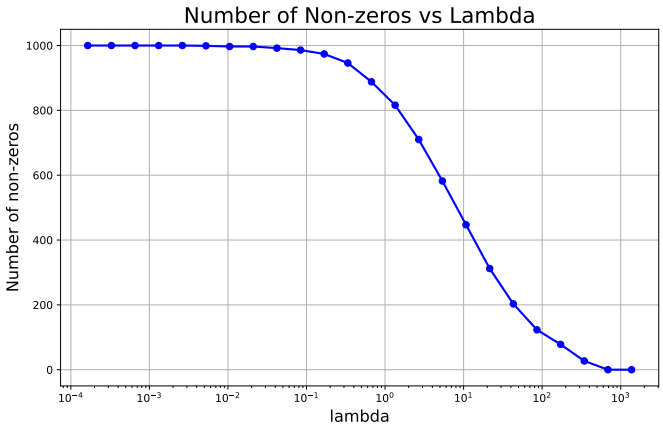



Figure 2: Number of Non-zeros vs Lambda

`Part b.` The plot is displayed in Figure 3.

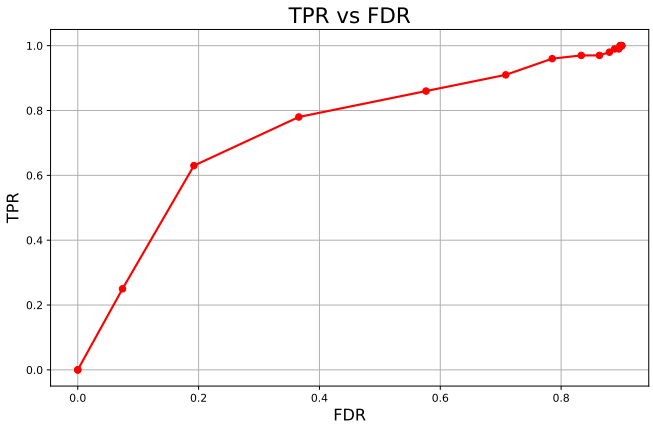



Figure 3: TPR vs FDR Plot

`Part c.` As $\lambda$ decreases from $\lambda_{max}$ to almost zero, we can see that
1) From Figure 2, the number of non-zero weights increases;
2) From Figure 3, both TPR and FDR generally increase.
It can be indicated that smaller $\lambda$ has better feature recovery (higher TPR) but at the cost of including more irrelevant features (higher FDR).

A6. We'll now put the Lasso to work on some real data in `crime_data_lasso.py`. We have read in the data for you with the following:

```
df_train, df_test = load_dataset("crime")
```

This stores the data as Pandas `DataFrame` objects. `DataFrame`s are similar to Numpy `array`s but more flexible; unlike `array`s, `DataFrame`s store row and column indices along with the values of the data. Each column of a `DataFrame` can also store data of a different type (here, all data are floats). Here are a few commands that will get you working with Pandas for this assignment:

```
df.head()                    # Print the first few lines of DataFrame df.
df.index                     # Get the row indices for df.
df.columns                   # Get the column indices.
df[''foo'']                 # Return the column named ''foo''.
df.drop(''foo'', axis = 1)  # Return all columns except ''foo''.
df.values                    # Return the values as a Numpy array.
df[''foo''].values          # Grab column foo and convert to Numpy array.
df.iloc[:3,:3]               # Use numerical indices (like Numpy) to get 3 rows and cols.
```

The data consist of local crime statistics for 1,994 US communities. The response $y$ is the rate of violent crimes reported per capita in a community. The name of the response variable is `ViolentCrimesPerPop`, and it is held in the first column of `df_train` and `df_test`. There are 95 features. These features include many variables. Some features are the consequence of complex political processes, such as the size of the police force and other systemic and historical factors. Others are demographic characteristics of the community, including self-reported statistics about race, age, education, and employment drawn from Census reports.

*The goals of this problem are threefold: (i) to encourage you to think about how data collection processes affect the resulting model trained from that data; (ii) to encourage you to think deeply about models you might train and how they might be misused; and (iii) to see how Lasso encourages sparsity of linear models in settings where $d$ is large relative to $n$.* **We emphasize that training a model on this dataset can suggest a degree of correlation between a community's demographics and the rate at which a community experiences and reports violent crime. We strongly encourage students to consider why these correlations may or may not hold more generally, whether correlations might result from a common cause, and what issues can result in misinterpreting what a model can explain.**

The dataset is split into a training and test set with 1,595 and 399 entries, respectively[1]. We will use this training set to fit a model to predict the crime rate in new communities and evaluate model performance on the test set. As there are a considerable number of input variables and fairly few training observations, overfitting is a serious issue. In order to avoid this, use the Lasso algorithm implemented in the previous problem.

a. *[4 points]* Read the documentation for the original version of this dataset: `http://archive.ics.uci.edu/ml/datasets/communities+and+crime`. Report 3 features included in this dataset for which historical *policy* choices in the US would lead to variability in these features. As an example, the *number of police* in a community is often the consequence of decisions made by governing bodies, elections, and amount of tax revenue available to decision makers.

b. *[4 points]* Before you train a model, describe 3 features in the dataset which might, if found to have nonzero weight in model, be interpreted as *reasons* for higher levels of violent crime, but which might actually be a *result* rather than (or in addition to being) the cause of this violence.

Now, we will run the Lasso solver. Begin with $\lambda = \lambda_{\max}$ defined in Equation (1). Initialize all weights to 0. Then, reduce $\lambda$ by a factor of 2 and run again, but this time initialize $\hat{w}$ from your $\lambda = \lambda_{\max}$ solution as your initial weights, as described above. Continue the process of reducing $\lambda$ by a factor of 2 until $\lambda < 0.01$. For all plots use a log-scale for the $\lambda$ dimension (Tip: use `plt.xscale('log')`).

[1]The features have been standardized to have mean 0 and variance 1.

c. *[4 points]* Plot the number of nonzero weights of each solution as a function of $\lambda$.

d. *[4 points]* Plot the regularization paths (in one plot) for the coefficients for input variables `agePct12t29`, `pctWSocSec`, `pctUrban`, `agePct65up`, and `householdsize`.

e. *[4 points]* On one plot, plot the squared error on the training and test data as a function of $\lambda$.

f. *[4 points]* Sometimes a larger value of $\lambda$ performs nearly as well as a smaller value, but a larger value will select fewer variables and perhaps be more interpretable. Retrain and inspect the weights $\hat{w}$ for $\lambda = 30$ and for *all* input variables. Which feature had the largest (most positive) Lasso coefficient? What about the most negative? Discuss briefly.

g. *[4 points]* Suppose there was a large negative weight on `agePct65up` and upon seeing this result, a politician suggests policies that encourage people over the age of 65 to move to high crime areas in an effort to reduce crime. What is the (statistical) flaw in this line of reasoning? (Hint: fire trucks are often seen around burning buildings, do fire trucks cause fire?)

## What to Submit:

- **Parts a, b:** 1-3 sentence explanation.
- **Part c:** Plot 1.
- **Part d:** Plot 2.
- **Part e:** Plot 3.
- **Parts f, g:** Answers and 1-2 sentence explanation.
- **Code** on Gradescope through coding submission.

`Part a.`
1) **PctUnemployed** (Percentage of people unemployed): Policies like trade agreements, infrastructure investment, and laws may cause unequal employment opportunities across different people.
2) **PctSpeakEnglOnly** (Percentage of people who speak only English): This is heavily influenced by immigration policies, bilingual education policies that affected which communities formed and thrived.
3) **PctImmigRecent** (Percentage of immigrants who immigated within last 3 years): Federal immigration policies, visa allocations, and border enforcement strategies that would vary across different people.

`Part b.`
1) **LemasTotReqPerPop** (Total requests for police per 100K population): A higher rate of police calls or requests could be interpreted as evidence of a population more prone to violence.
2) **PolicCars** (Number of police cars): A place with many police cars is more likely a reaction to existing crime problems than a cause.
3) **PctVacMore6Mos** (Percent of vacant housing that has been vacant more than 6 months): The percentage of vacant properties might appear to cause crime by providing spaces for illegal activities, but high vacancy rates may be the result of violence.

`Part c.` The plot is shown in Figure 4.

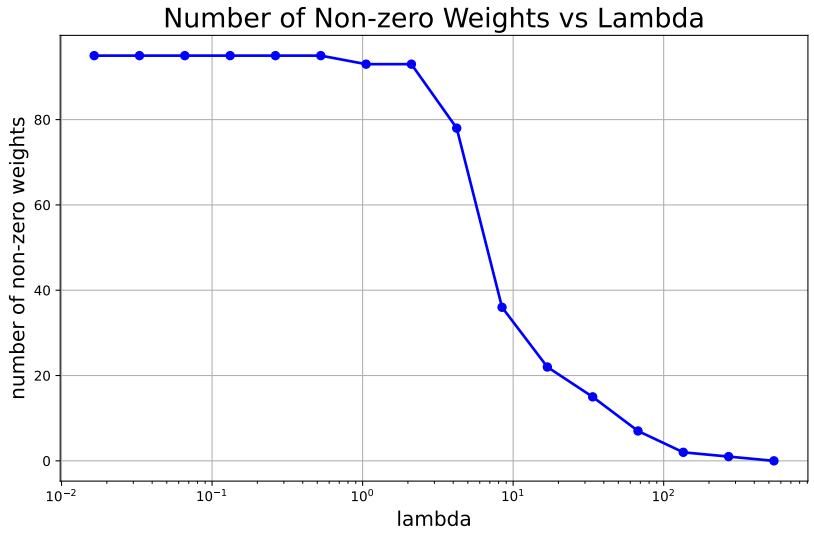



Figure 4: Number of Nonzero Weights of Each $\lambda$

`Part d.` The plot is displayed in Figure 5.

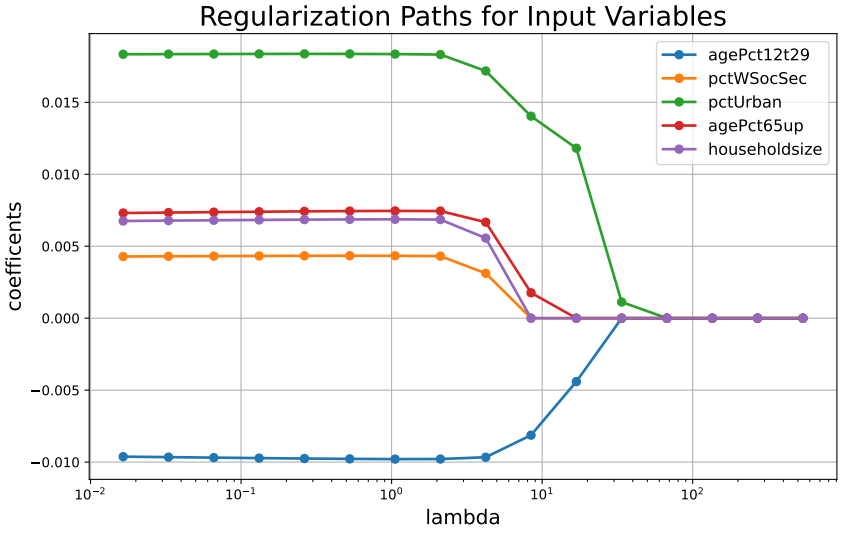



Figure 5: Regularization Paths for the Coefficients for Input Variables

`Part e.` The plot is shown in Figure 6.

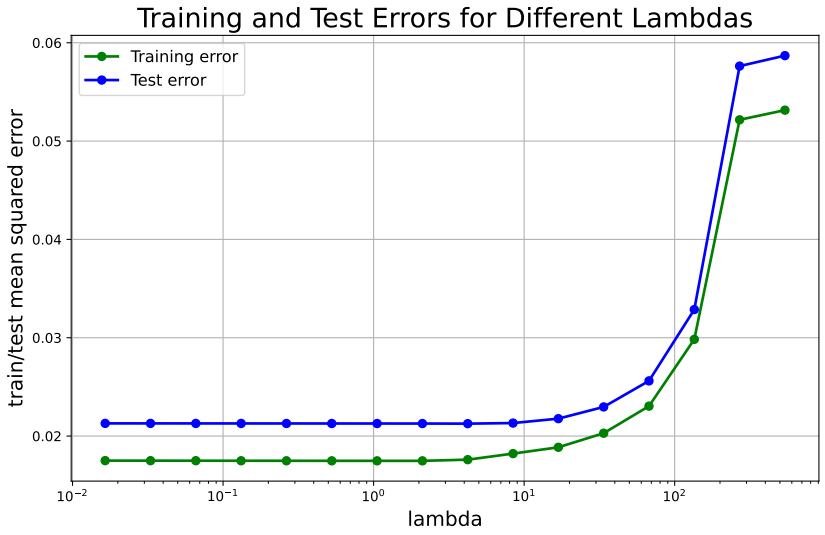



Figure 6: Training and Test Error for Different $\lambda$

`Part f.`
The most positive feature is **PctIlleg**, with coefficient 0.0698. This feature is the percentage of children born to unmarried parents. The positive coefficient suggests that communities with higher rates of children born to unmarried parents tend to have higher violent crime rates.
The most negative feature is **PctKids2Par**, with coefficient $-0.0650$. This feature represents the percentage of kids living with two parents. The negative coefficient indicates that communities with more children living in two-parent households tend to have lower violent crime rates.

`Part g.`
The statistical flaw is that correlation does not imply causation.
A negative coefficient might indicate that areas with more elderly residents have less crime, but this doesn't mean that moving elderly people to high-crime areas will reduce crime.
Elderly people may prefer to live in safer areas. Areas with certain characteristics, like perfect medical care, or sufficient parks or gardens, would also attract elderly resident to live in.

# Logistic Regression

## Binary Logistic Regression

A7. Here we consider the MNIST dataset, but for binary classification. Specifically, the task is to determine whether a digit is a 2 or 7. Here, let $Y = 1$ for all the "7" digits in the dataset, and use $Y = -1$ for "2". We will use regularized logistic regression. Given a binary classification dataset $\{(x_i, y_i)\}_{i=1}^n$ for $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$ we showed in class that the regularized negative log likelihood objective function can be written as

$$J(w, b) = \frac{1}{n} \sum_{i=1}^{n} \log(1 + \exp(-y_i(b + x_i^T w))) + \lambda ||w||_2^2$$

Note that the offset term $b$ is not regularized. For all experiments, use $\lambda = 10^{-1}$. Let $\mu_i(w, b) = \frac{1}{1+\exp(-y_i(b+x_i^T w))}$.

a. *[8 points]* Derive the gradients $\nabla_w J(w, b)$, $\nabla_b J(w, b)$ and give your answers in terms of $\mu_i(w, b)$ (your answers should not contain exponentials).

b. *[8 points]* Implement gradient descent with an initial iterate of all zeros. Try several values of step sizes to find one that appears to make convergence on the training set as fast as possible. Run until you feel you are near to convergence.

   (i) For both the training set and the test, plot $J(w, b)$ as a function of the iteration number (and show both curves on the same plot).

   (ii) For both the training set and the test, classify the points according to the rule $\text{sign}(b + x_i^T w)$ and plot the misclassification error as a function of the iteration number (and show both curves on the same plot).

   Reminder: Make sure you are only using the test set for evaluation (not for training).

c. *[7 points]* Repeat (b) using stochastic gradient descent with a batch size of 1. Note, the expected gradient with respect to the random selection should be equal to the gradient found in part (a). Show both plots described in (b) when using batch size 1. Take careful note of how to scale the learning rate.

d. *[7 points]* Repeat (b) using stochastic gradient descent with batch size of 100. That is, instead of approximating the gradient with a single example, use 100. Note, the expected gradient with respect to the random selection should be equal to the gradient found in part (a).

## What to Submit

- **Part a:** Proof
- **Part b:** Separate plots for b(i) and b(ii).
- **Part c:** Separate plots for c which reproduce those from b(i) and b(ii) for this case.
- **Part d:** Separate plots for c which reproduce those from b(i) and b(ii) for this case.
- **Code** on Gradescope through coding submission.

`Part a.` Solve.
Let $s_i = y_i(b + x_i^T w)$, then

$$\frac{\partial s_i}{\partial w} = y_i x_i, \quad \frac{\partial s_i}{\partial b} = y_i.$$

Now the derivative of $\log(1 + \exp(-y_i(b + x_i^T w)))$ with respect to $s_i$ is

$$\frac{d}{ds_i} \log(1 + \exp(-y_i(b + x_i^T w))) = \frac{d}{ds_i} \log(1 + e^{-s_i}) = \frac{-e^{-s_i}}{1 + e^{-s_i}}$$

Note that $\mu_i = \frac{1}{1+e^{-s_i}}$, so $1 - \mu_i = \frac{e^{s_i}}{1+e^{-s_i}}$.
Then

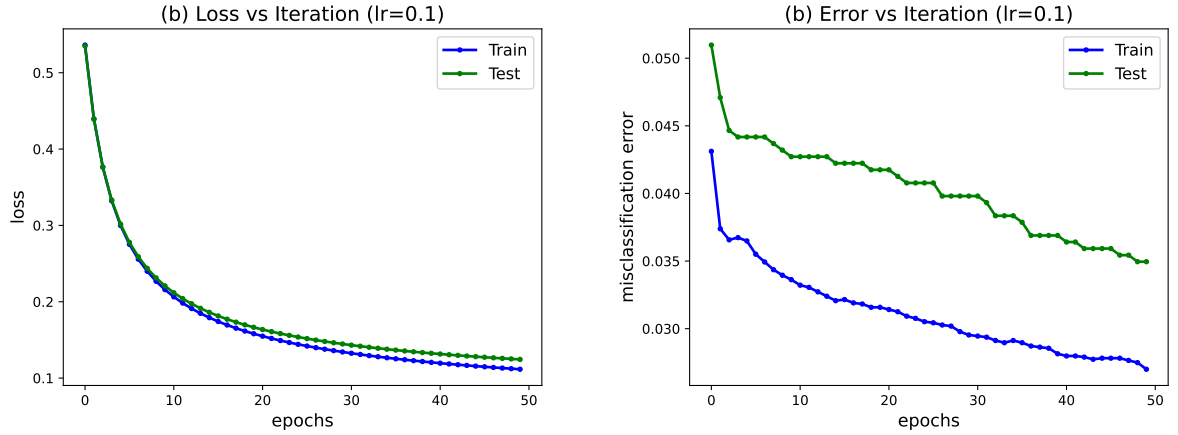
$$\frac{d}{ds_i} \log(1 + \exp(-y_i(b + x_i^T w))) = \mu_i - 1$$

Thus

$$\nabla_w J(w, b) = \frac{1}{n} \sum_{i=1}^{n} \frac{\partial s_i}{\partial w} \frac{d}{ds_i} \log(1 + e^{-s_i}) + \frac{d}{dw}(\lambda w^T w) = \frac{1}{n} \sum_{i=1}^{n} \big(\mu_i(w, b) - 1\big) y_i x_i + 2\lambda w$$

$$\nabla_b J(w, b) = \frac{1}{n} \sum_{i=1}^{n} \frac{\partial s_i}{\partial b} \frac{d}{ds_i} \log(1 + e^{-s_i}) = \frac{1}{n} \sum_{i=1}^{n} \big(\mu_i(w, b) - 1\big) y_i$$

`Part b.` Separate plots are shown in Figure 7.

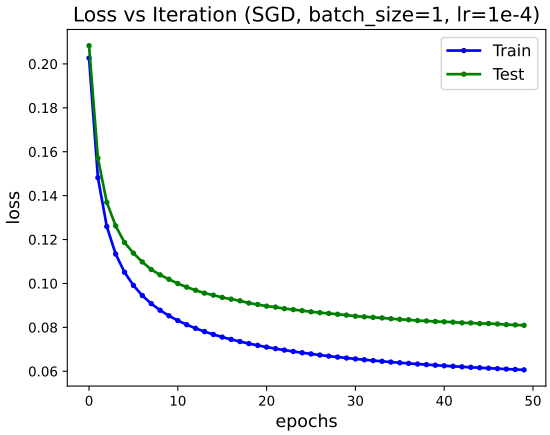



(a) Training/Test Loss vs Iteration Number

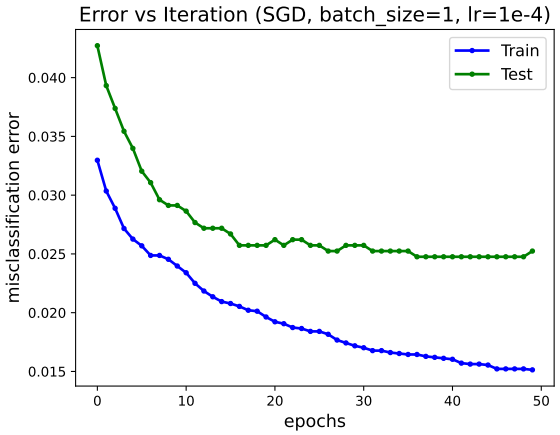
(b) Misclassification Error vs Iteration Number

Figure 7: (b) Gradient Descent with Initial Iterate

`Part c.` Separate plots are shown in Figure 8.



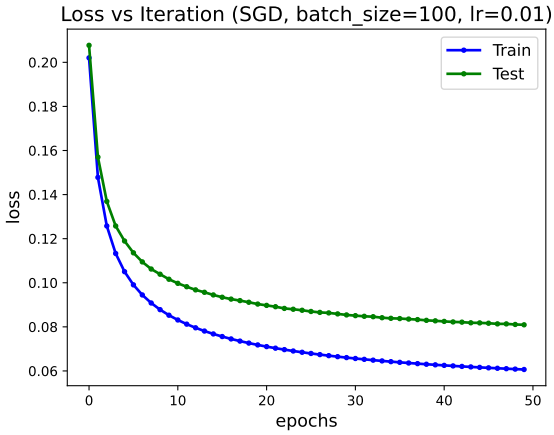

(a) Training/Test Loss vs Iteration Number



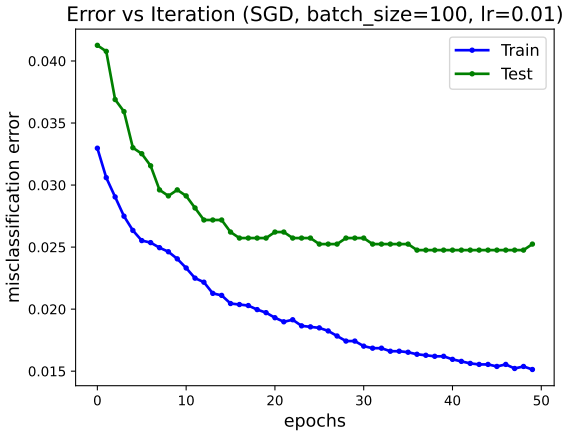

(b) Misclassification Error vs Iteration Number

Figure 8: (c) Stocastic Gradient Descent with Batch Size of 1

`Part d.` Separate plots are shown in Figure 9.




(a) Training/Test Loss vs Iteration Number




(b) Misclassification Error vs Iteration Number

Figure 9: (d) Stocastic Gradient Descent with Batch Size of 100

# Confidence Interval of Least Squares Estimation

## Bounding the Estimate

B2. Let us consider the setting, where we have $n$ inputs, $X_1, ..., X_n \in \mathbb{R}^d$, and $n$ observations $Y_i = \langle X_i, \beta^* \rangle + \epsilon_i$, for $i = 1, ..., n$. Here, $\beta^*$ is a ground truth vector in $\mathbb{R}^d$ that we are trying to estimate, the noise $\epsilon_i \sim \mathcal{N}(0, 1)$, and the $n$ examples piled up — $X \in R^{n \times d}$. To estimate, we use the least squares estimator $\widehat{\beta} = \min_\beta \|X\beta - Y\|_2^2$. Moreover, we will use $n = 20000$ and $d = 10000$ in this problem.

a. *[3 points]* Show that $\widehat{\beta}_j \sim \mathcal{N}(\beta_j^*, (X^T X)^{-1}_{j,j})$ for each $j = 1, ..., d$. *(Hint: see notes on confidence intervals.)*

b. *[4 points]* Fix $\delta \in (0, 1)$ suppose $\beta^* = 0$. Applying the proposition from the notes, conclude that for each $j \in [d]$, with probability at least $1 - \delta$, $|\widehat{\beta}_j| \leq \sqrt{2(X^T X)^{-1}_{j,j} \log(2/\delta)}$. Can we conclude that with probability at least $1 - \delta$, $|\widehat{\beta}_j| \leq \sqrt{2(X^T X)^{-1}_{j,j} \log(2/\delta)}$ for all $j \in [d]$ simultaneously? Why or why not?

c. *[5 points]* Let's explore this question empirically. Assume data is generated as $x_i = \sqrt{(i \mod d) + 1} \cdot e_{(i \mod d)+1}$ where $e_i$ is the $i$th canonical vector and $i \mod d$ is the remainder of $i$ when divided by $d$. Generate each $y_i$ according to the model above. Compute $\widehat{\beta}$ and plot each $\widehat{\beta}_j$ as a scatter plot with the $x$-axis as $j \in \{1, \ldots, d\}$. Plot $\pm\sqrt{2(X^T X)^{-1}_{j,j} \log(2/\delta)}$ as the upper and lower confidence intervals with $1 - \delta = 0.95$. How many $\widehat{\beta}_j$'s are outside the confidence interval? *Hint: Due to the special structure of how we generated $x_i$, we can compute $(X^T X)^{-1}$ analytically without computing an inverse explicitly.*

### What to Submit:

- **Parts a, b:** Proof.
- **Part b:** Answer.
- **Part c:** Plots of $\hat{\beta}$ and its confidence interval **on the same plot**.

`Part a.` Proof.
From the least square estimator $\widehat{\beta} = \arg\min_\beta \|X\beta - Y\|_2^2$, we have

$$f = \|X\beta - Y\|_2^2 = (X\beta - Y)^\top (X\beta - Y) = \beta^\top X^\top X\beta - 2Y^\top X\beta + Y^\top Y.$$

Now we differentiate with respect to $\beta$ and set the derivative to zero,

$$\frac{\partial f}{\partial \beta} = 2X^\top X\beta - 2X^\top Y = 0 \quad \Rightarrow \quad X^\top X\beta = X^\top Y \quad \Rightarrow \quad \widehat{\beta} = (X^\top X)^{-1}X^\top Y$$

Since $Y = X\beta^* + \varepsilon$, we get

$$\widehat{\beta} = (X^\top X)^{-1}X^\top Y = (X^\top X)^{-1}X^\top (X\beta^* + \varepsilon) = \beta^* + (X^\top X)^{-1}X^\top \varepsilon.$$

Since $\varepsilon \sim \mathcal{N}(0, I_n)$, applying *Proposition 1* in the notes with $A = (X^T X)^{-1}X^T$ and $b = \beta^*$, we have

$$\widehat{\beta} \sim \mathcal{N}\big(\beta^*,\ (X^\top X)^{-1}X^\top I_n X(X^\top X)^{-1}\big) = \mathcal{N}\big(\beta^*,\ (X^\top X)^{-1}\big).$$

Therefore, take the $j$-th coordinate yields

$$\widehat{\beta}_j \sim \mathcal{N}\big(\beta_j^*,\ (X^\top X)^{-1}_{j,j}\big),\ \text{for each } j = 1, 2, \ldots, d$$

`Part b.` Proof.
Since for each $j = 1, 2, \ldots, d$, $\widehat{\beta}_j \sim \mathcal{N}\big(\beta_j^*,\ (X^\top X)^{-1}_{j,j}\big)$, $\widehat{\beta}$ is Gaussian with mean 0 ($\beta^* = 0$) and covariance matrix $(X^\top X)^{-1}_{j,j}$.
Let $\sigma_j^2 = \big((X^\top X)^{-1}\big)_{jj}$, then $\widehat{\beta}_j/\sigma_j \sim \mathcal{N}(0,1)$.
By *Proposition 2* from the notes, for any fixed $j \in [d]$, for some $\delta \in (0,1)$,

$$\mathbb{P}\Big(|\widehat{\beta}_j| \le \sqrt{2\sigma_j^2 \log(2/\delta)}\Big) = 1 - \mathbb{P}\Big(|\widehat{\beta}_j| > \sqrt{2\sigma_j^2 \log(2/\delta)}\Big) \ge 1 - \delta$$

where $\mathbb{P}\Big(|\widehat{\beta}_j| > \sqrt{2\sigma_j^2 \log(2/\delta)}\Big) \le \delta$.
Let $A_j$ be the event that with probability at least $1-\delta$, $|\widehat{\beta}_j| \le \sqrt{2(X^T X)^{-1}_{j,j} \log(2/\delta)}$, then for each $j$, $\mathbb{P}(A_j) \ge 1 - \delta$ does not imply $\mathbb{P}\big(\bigcap_{j=1}^d A_j\big) \ge 1 - \delta$ for all $j$. This is because the probability can be highly dependent across j.
We know that

$$\mathbb{P}\Big(\exists j \in [d] \text{ s.t. } |\widehat{\beta}_j| > t\Big) \le \sum_{j=1}^d \mathbb{P}\big(|\widehat{\beta}_j| > t\big).$$

If we choose $t = \sqrt{2\sigma_j^2 \log(2/\delta)}$ ($t$ may be different across $j$), each term inside the summation is at most $\delta$, and then we obtain

$$\mathbb{P}\Big(\exists j \in [d] \text{ with } |\widehat{\beta}_j| > t_j\Big) \le d\delta,$$

$$\Rightarrow \mathbb{P}\Big(\forall j \in [d] :\ |\widehat{\beta}_j| \le t_j\Big) \ge 1 - d\delta.$$

Thus we cannot guarantee the event holds for all $j$ with probability $1-\delta$ unless $d\delta \le \delta \Rightarrow d \le 1$.

`Part c.` Solve. The plot is displayed in Figure 10.

There are 72 $\widehat{\beta}_j$s outside the confidence interval.

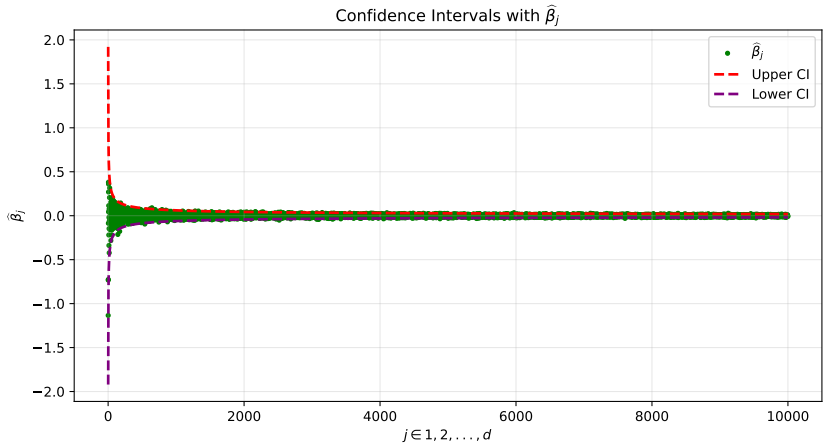



Figure 10: Confidence Intervals with $\widehat{\beta}_j$