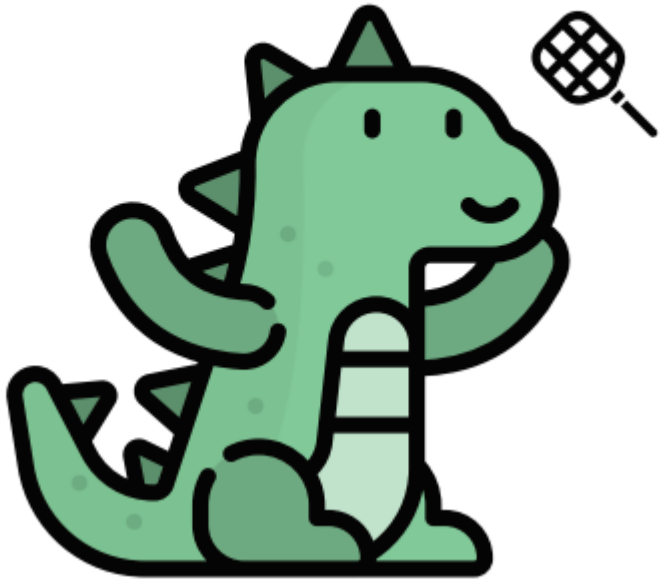
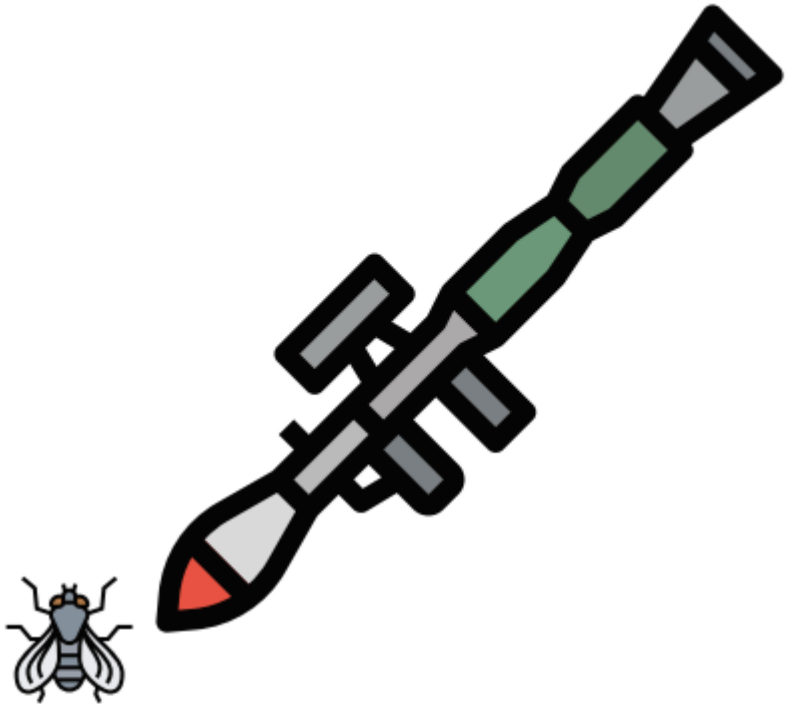
Optimizing the training process

Underfitting, overfitting, testing, and regularization


USER FRIENDLY by J.D. "Illiad" Frazer
I WENT FOR LUNCH AND WHEN I CAME BACK MY MODEL HAD OVERFIT!
COPYRIGHT©2008 J.D. "Illiad" Frazer HTTP://WWW.USERFRIENDLY.ORG/
OH NO! WHAT ARE YOU GOING TO DO?
I'M NEVER GOING FOR LUNCH AGAIN.
GENIUS.



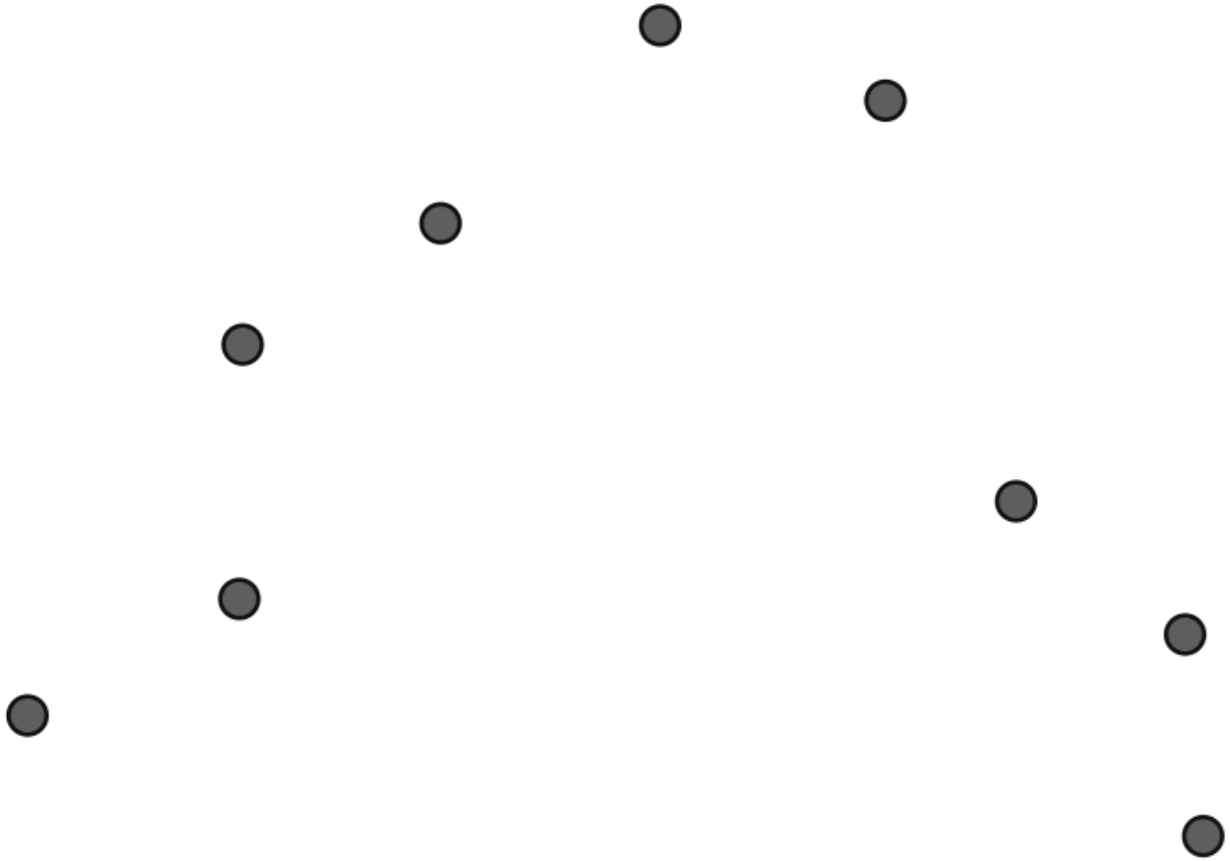
Underfitting



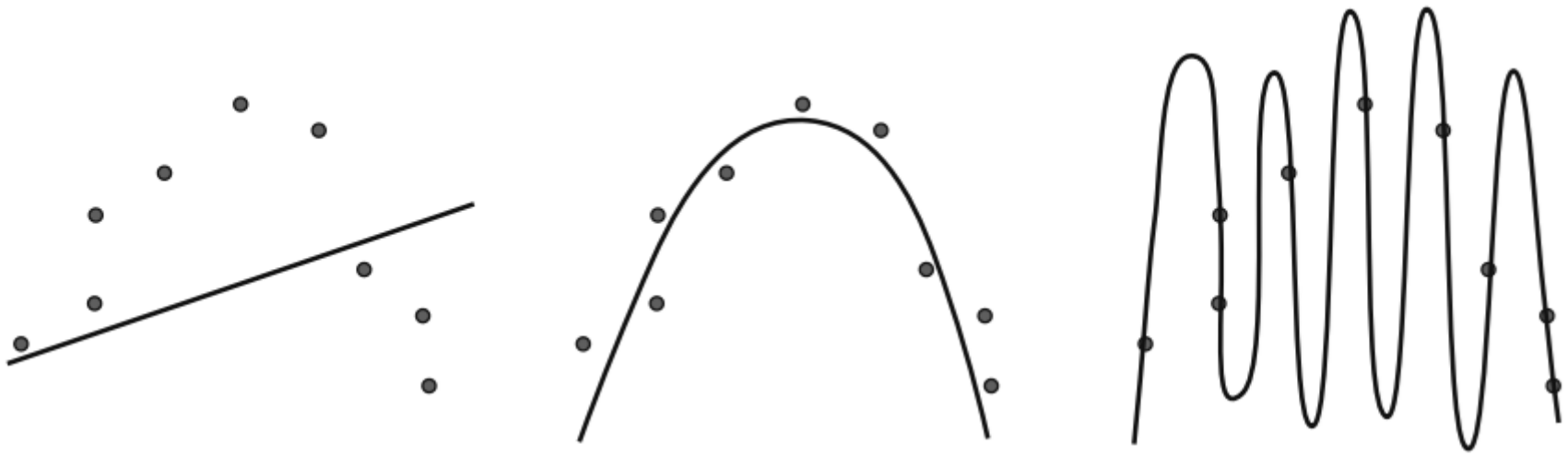
Overfitting

Underfitting and overfitting are two problems that can occur when training our machine learning model. Left: Underfitting happens when we oversimplify the problem at hand, and we try to solve it using a simple solution, such as trying to kill Godzilla using a fly swatter. Right: Overfitting happens when we overcomplicate the solution to a problem and try to solve it with an exceedingly complicated solution, such as trying to kill a fly using a bazooka.

An example



In this dataset, we train some models and exhibit training problems such as underfitting and overfitting. If you were to fit a polynomial regression model to this dataset, what type of polynomial would you use: a line, a parabola, or something else?



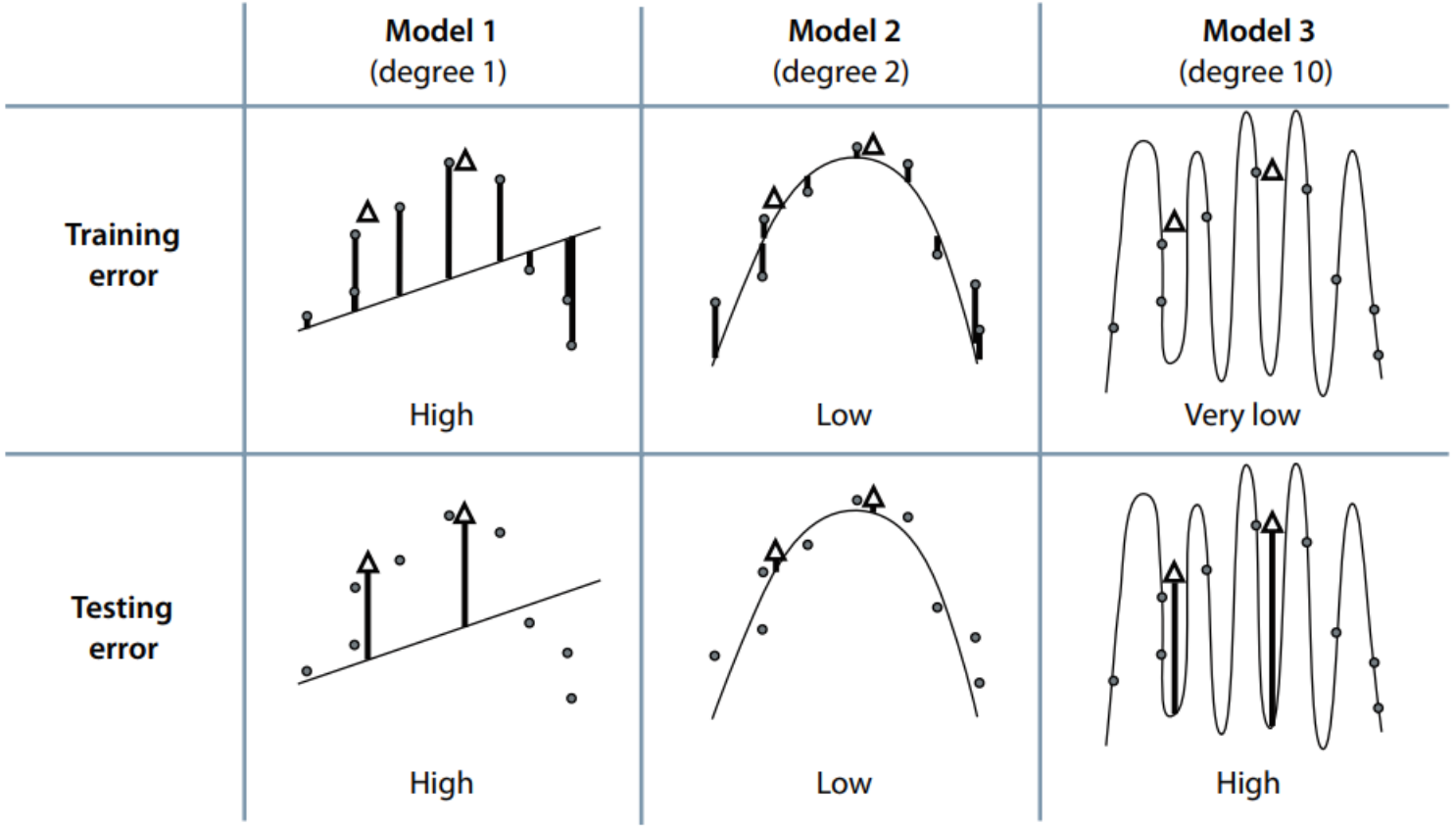
Model 1
Polynomial of degree 1
(a line)

Model 2
Polynomial of degree 2
(a parabola)

Model 3
Polynomial of degree 10

Fitting three models to the same dataset. Model 1 is a polynomial of degree 1, which is a line. Model 2 is a polynomial of degree 2, or a quadratic. Model 3 is a polynomial of degree 10. Which one looks like the best fit?

How do we get the computer to pick the right model? By testing



Summary

Models can

- Underfit: use a model that is too simple to our dataset.
- Fit the data well: use a model that has the right amount of complexity for our dataset.
- Overfit: use a model that is too complex for our dataset.

In the training set

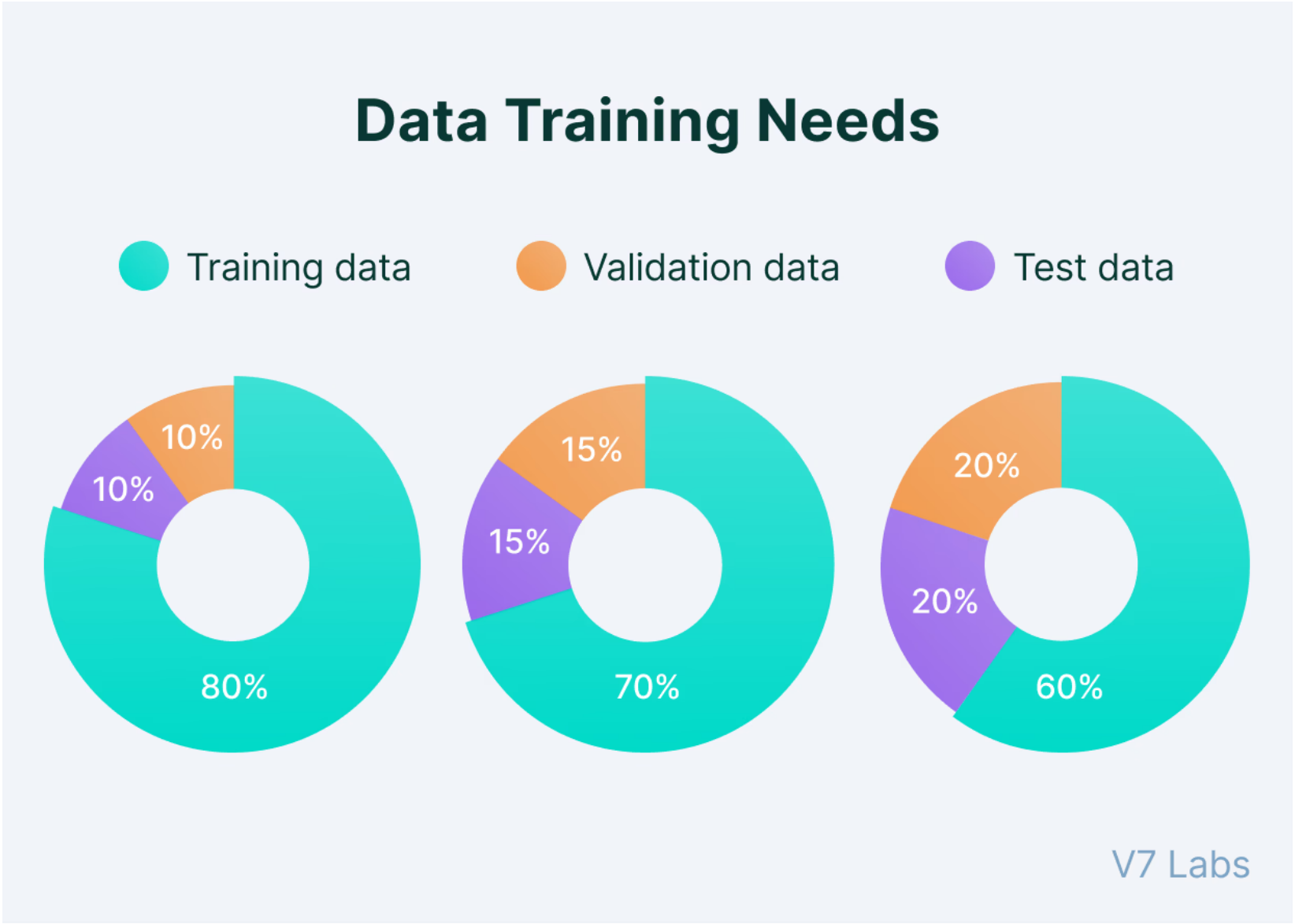
- The underfit model will do poorly (large training error).
- The good model will do well (small training error).
- The overfit model will do very well (very small training error).

In the testing set

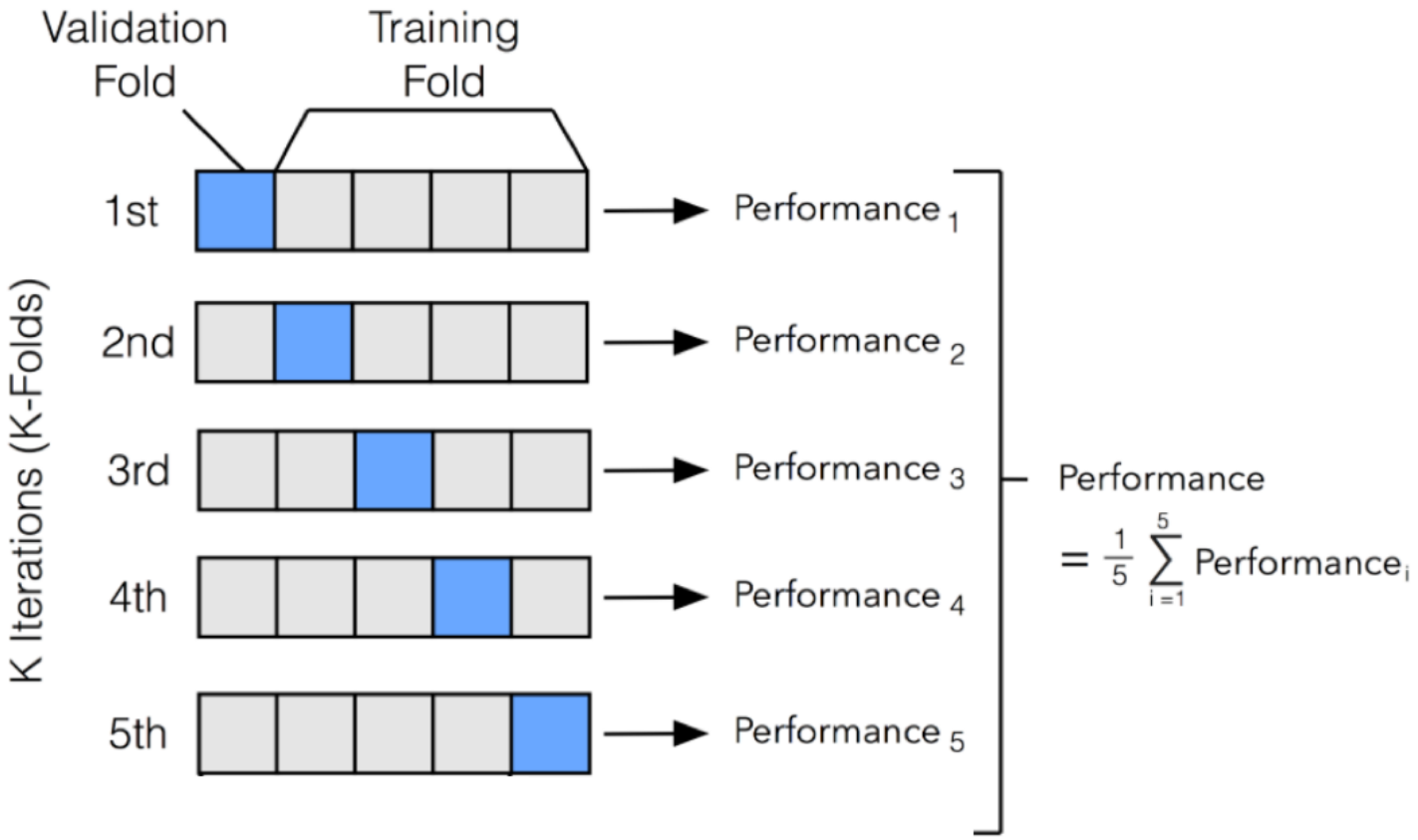
- The underfit model will do poorly (large testing error).
- The good model will do well (small testing error).
- The overfit model will do poorly (large testing error).

Thus, the way to tell whether a model underfits, overfits, or is good, is to look at the training and testing errors. If both errors are high, then it underfits. If both errors are low, then it is a good model. If the training error is low and the testing error is high, then it overfits.

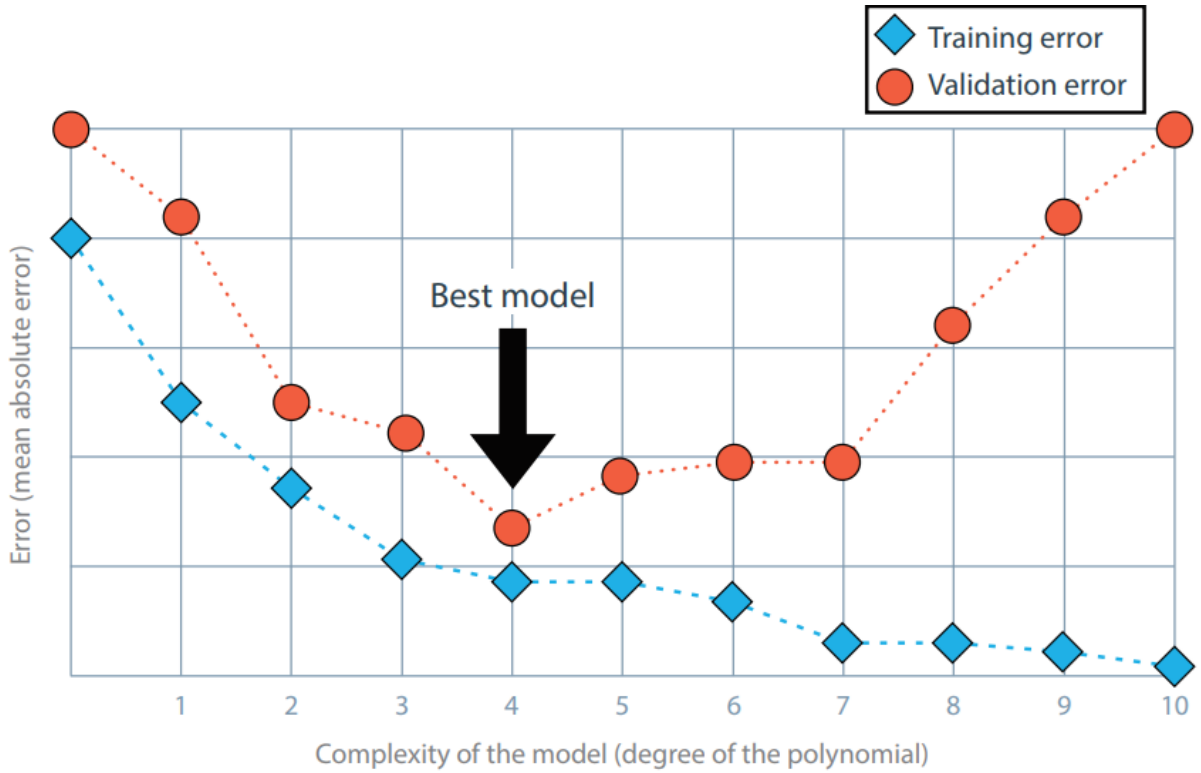
golden rule Thou shalt never use your testing data for training.


Data Training Needs
Training data
Validation data
Test data
10%
10%
80%
15%
15%
70%
20%
20%
60%
V7 Labs

K-Fold Cross Validation



A numerical way to decide how complex our model should be: The model complexity graph



The model complexity graph is an effective tool to help us determine the ideal complexity of a model to avoid underfitting and overfitting.

Another alternative to avoiding overfitting: Regularization

- Another useful technique to avoid overfitting in our models that doesn't require a testing set: *regularization*.
- Regularization can help:
  - Don't need to train several models to select the one that best balanced performance and complexity.
  - Simply train the model once, but during the training, we try to not only improve the model's performance but also reduce its complexity.
  - The key to doing this is to measure both performance and complexity at the same time.

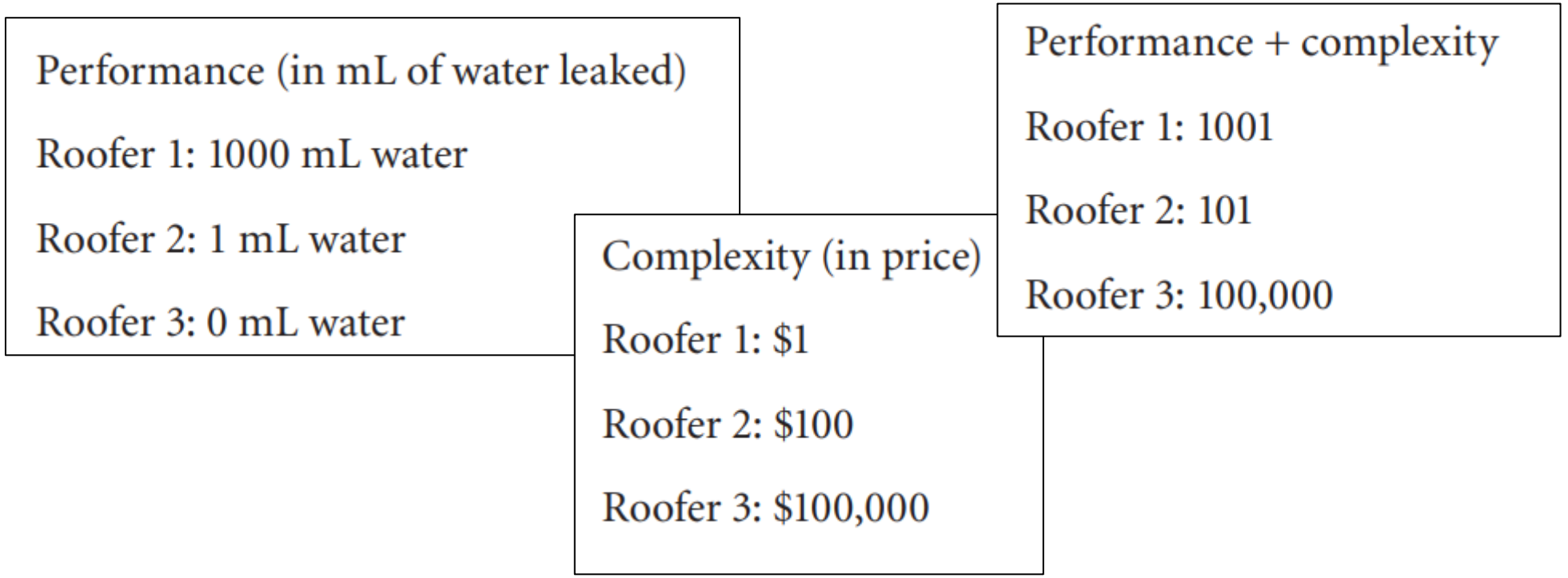
Problem: Broken roof

Roofer 1
Solution: Bandage
(Underfitting)

Roofer 2
Solution: Shingles
(Correct)

Roofer 3
Solution: Titanium
(Over!tting)

An analogy for underfitting and overfitting. Our problem consists of a broken roof. We have three roofers who can fix it. Roofer 1 comes with a bandage, roofer 2 comes with roofing shingles, and roofer 3 comes with a block of titanium. Roofer 1 oversimplified the problem, so they represent underfitting. Roofer 2 used a good solution. Roofer 3 overcomplicated the solution, so they represent overfitting.



- Now it is clear that roofer 2 is the best one, which means that optimizing performance and complexity at the same time yields good results that are also as simple as possible.
- This is what regularization is about: measuring performance and complexity with two different error functions, and adding them to get a more robust error function.
- This new error function ensures that our model performs well and is not very complex.

Another example of overfitting: Movie recommendations

Imagine that we have a movie streaming website, and we are trying to build a recommender system. For simplicity, imagine that we have only 10 movies: M1, M2, …, M10. A new movie, M11, comes out, and we'd like to build a linear regression model to recommend movie 11 based on the previous 10. We have a dataset of 100 users. For each user, we have 10 features, which are the times (in seconds) that the user has watched each of the original 10 movies. If the user hasn't watched a movie, then this amount is 0. The label for each user is the amount of time the user watched movie 11. We want to build a model that fits this dataset.

$$\hat{y} = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6 + w_7x_7 + w_8x_8 ++ w_9x_9 + w_{10}x_{10} + b,$$

where

- $\hat{y}$ is the amount of time the model predicts that the user will watch movie 11,
- x_i is the amount of time the user watched movie i, for $i = 1, 2, \ldots, 10$,
- w_i is the weight associated to movie i, and
- b is the bias.

Now let's test our intuition. Out of the following two models (given by their equation), which one (or ones) looks like it may be overfitting?

Model 1: $\hat{y} = 2x_3 + 1.4x_7 - 0.5x_7 + 4$

Model 2: $\hat{y} = 22x_1 - 103x_2 - 14x_3 + 109x_4 - 93x_5 + 203x_6 + 87x_7 - 55x_8 + 378x_9 - 25x_{10} + 8$

Measuring how complex a model is: L1 and L2 norm

- *L1 norm*: The sum of the absolute values of the coefficients.
- *L2 norm*: The sum of the squares of the coefficients.
- *The bias in the models is not included in the L1 and L2 norm:*
  - *The bias in the model is precisely the number of seconds that we expect a user to watch movie 11 if they haven't watched any of the previous 10 movies.*
  - *This number is not associated with the complexity of the model.*
- They come from a more general theory of L^P spaces, named after the French mathematician Henri Lebesgue.
- *Absolute values and squares are used to get rid of the negative coefficients; otherwise, large negative numbers will cancel out with large positive numbers, and we may end up with a small value for a very complex model.*

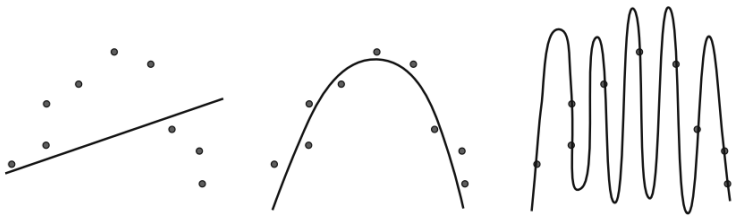
L1 norm:

- **Model 1**: $|2| + |1.4| + |-0.5| = 3.9$
- **Model 2**: $|22| + |-103| + |-14| + |109| + |-93| + |203| + |87| + |-55| + |378| + |-25| = 1{,}089$

L2 norm:

- **Model 1**: $2^2 + 1.4^2 + (-0.5)^2 = 6.21$
- **Model 2**: $22^2 + (-103)^2 + (-14)^2 + 109^2 + (-93)^2 + 203^2 + 87^2 + (-55)^2 + 378^2 + (-25)^2 = 227{,}131$

As expected, both the L1 and L2 norms of model 2 are much larger than the corresponding norms of model 1.



Model 1
Polynomial of degree 1
(a line)

Model 2
Polynomial of degree 2
(a parabola)

Model 3
Polynomial of degree 10

- **Model 1:** $\hat{y} = 2x + 3$
- **Model 2:** $\hat{y} = -x^2 + 6x - 2$
- **Model 3:** $\hat{y} = x^9 + 4x^8 - 9x^7 + 3x^6 - 14x^5 - 2x^4 - 9x^3 + x^2 + 6x + 10$

The L1 and L2 norms are calculated as follows:

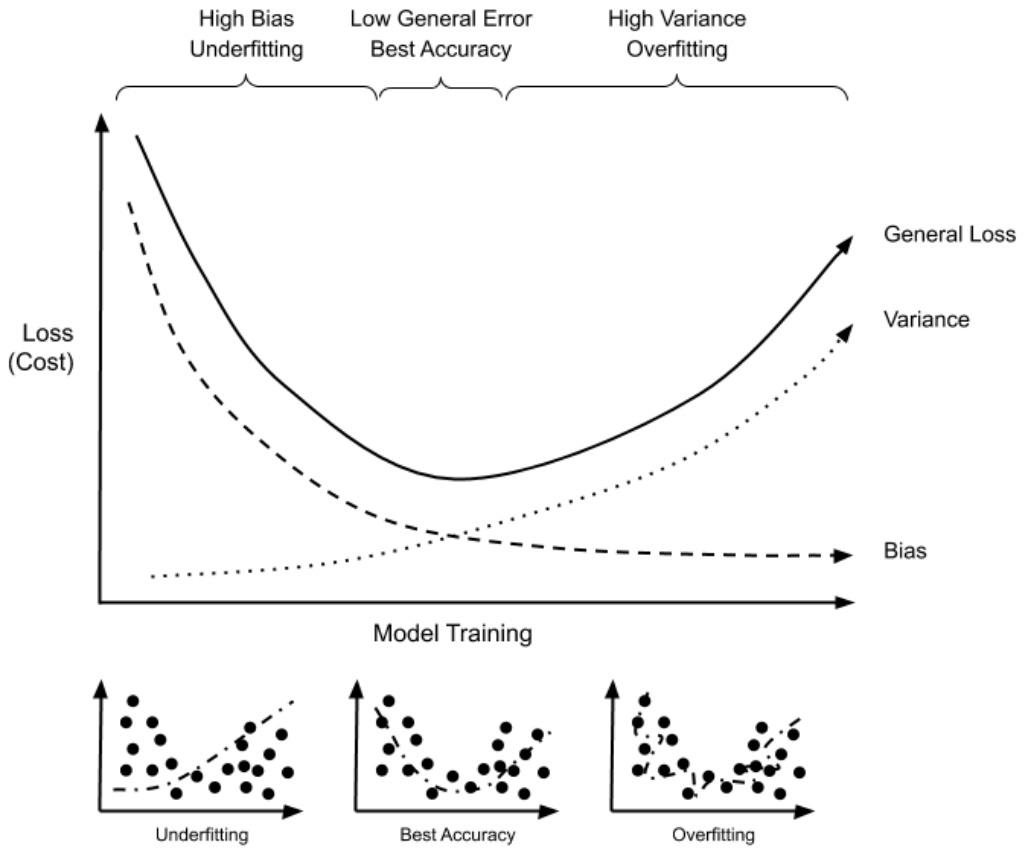
L1 norm:

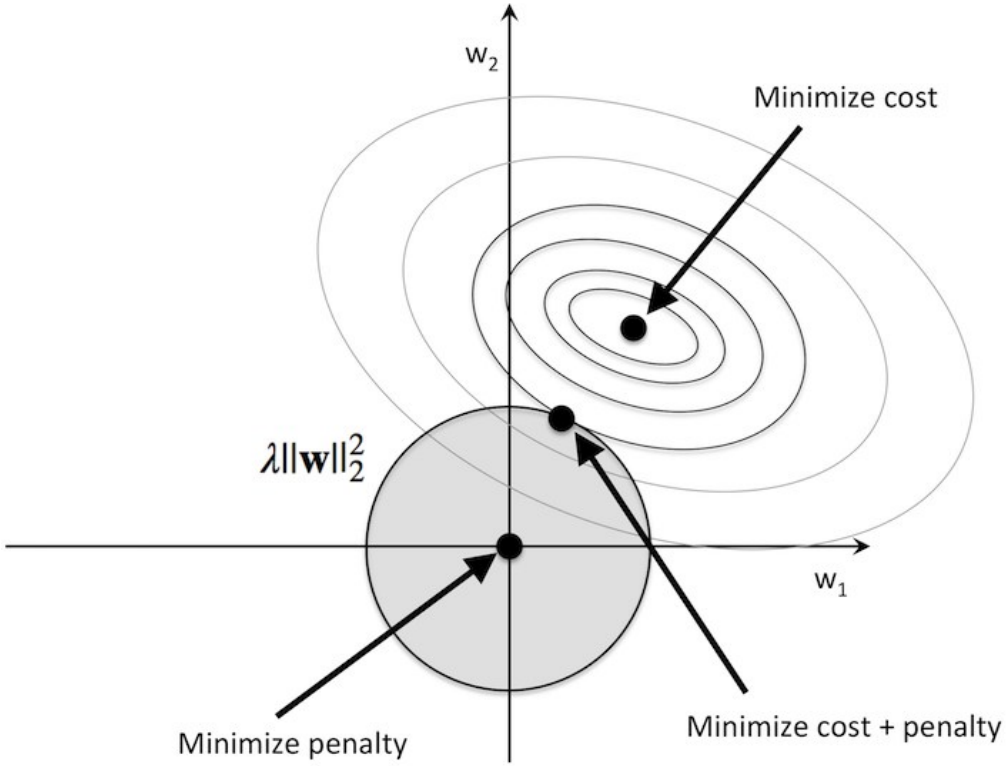
- **Model 1:** $|2| = 2$
- **Model 2:** $|-1| + |6| = 7$
- **Model 3:** $|1| + |4| + |-9| + |3| + |-14| + |-2| + |-9| + |1| + |6| = 49$

L2 norm:

- **Model 1:** $2^2 = 2$
- **Model 2:** $(-1)^2 + 6^2 = 37$
- **Model 3:** $1^2 + 4^2 + (-9)^2 + 3^2 + (-14)^2 + (-2)^2 + (-9)^2 + 1^2 + 6^2 = 425$

Modifying the error function to solve our problem: Lasso regression and ridge regression





regression error A measure of the quality of the model.

regularization term A measure of the complexity of the model. It can be the L1 or the L2 norm of the model.

The quantity that we want to minimize to find a good and not too complex model is the modified error, defined as the sum of the two, as shown next:

$$\text{Error} = \text{Regression error} + \text{Regularization term}$$

If we train our regression model using the L1 norm, the model is called *lasso regression*. Lasso stands for "least absolute shrinkage and selection operator." The error function follows:

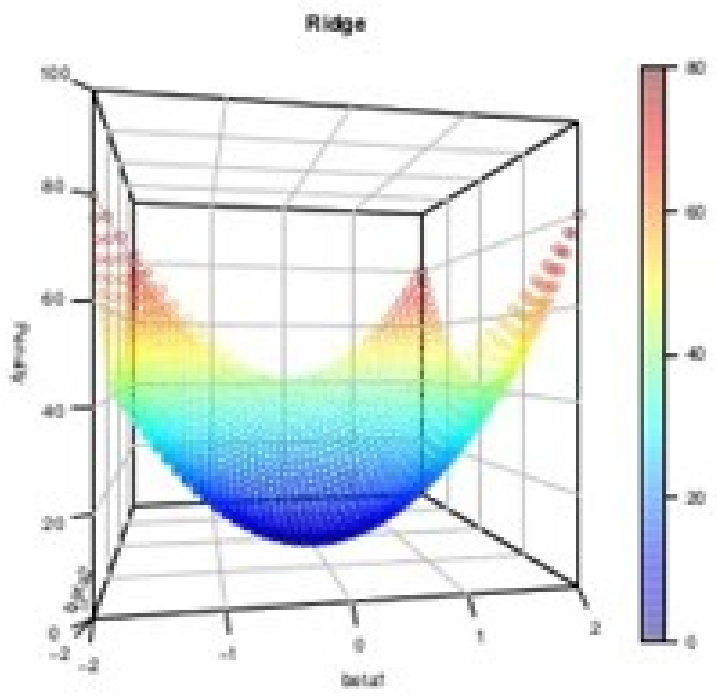
$$\text{Lasso regression error} = \text{Regression error} + \text{L1 norm}$$

If, instead, we train the model using the L2 norm, it is called *ridge regression*. The name *ridge* comes from the shape of the error function, because adding the L2 norm term to the regression error function turns a sharp corner into a smooth valley when we plot it. The error function follows:

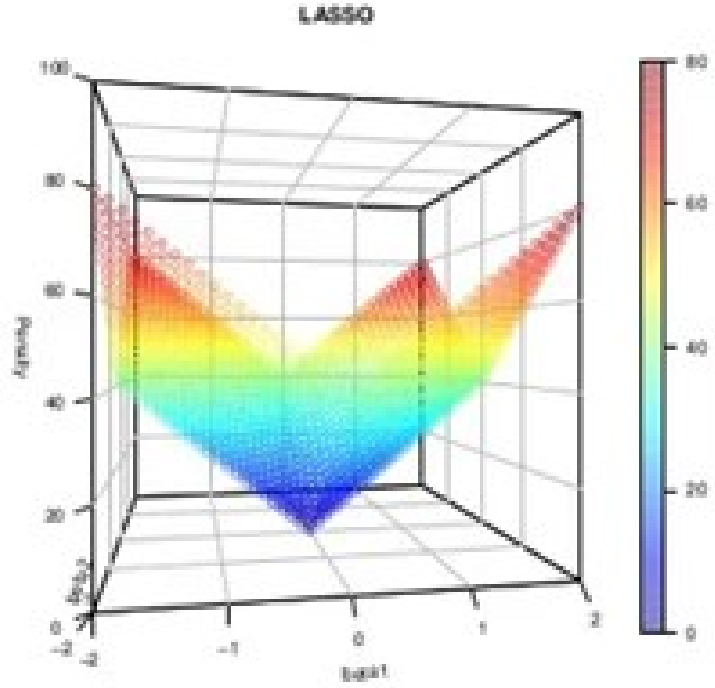
$$\text{Ridge regression error} = \text{Regression error} + \text{L2 norm}$$

Penalty Functions

Ridge Regression



LASSO



The regularization parameter

- A hyperparameter (λ) is called the *regularization parameter*, and its goal is to determine if the model-training process should emphasize performance or simplicity.

$$\text{Error} = \text{Regression error} + \lambda \text{ Regularization term}$$

- Picking a value of 0 for λ cancels out the regularization term.
- Picking a large value for λ results in a simple model, perhaps of low degree, which may not fit our dataset very well.
- It is typical to choose λ as powers of 10, such as 10, 1, 0.1, 0.01, but this choice is somewhat arbitrary.

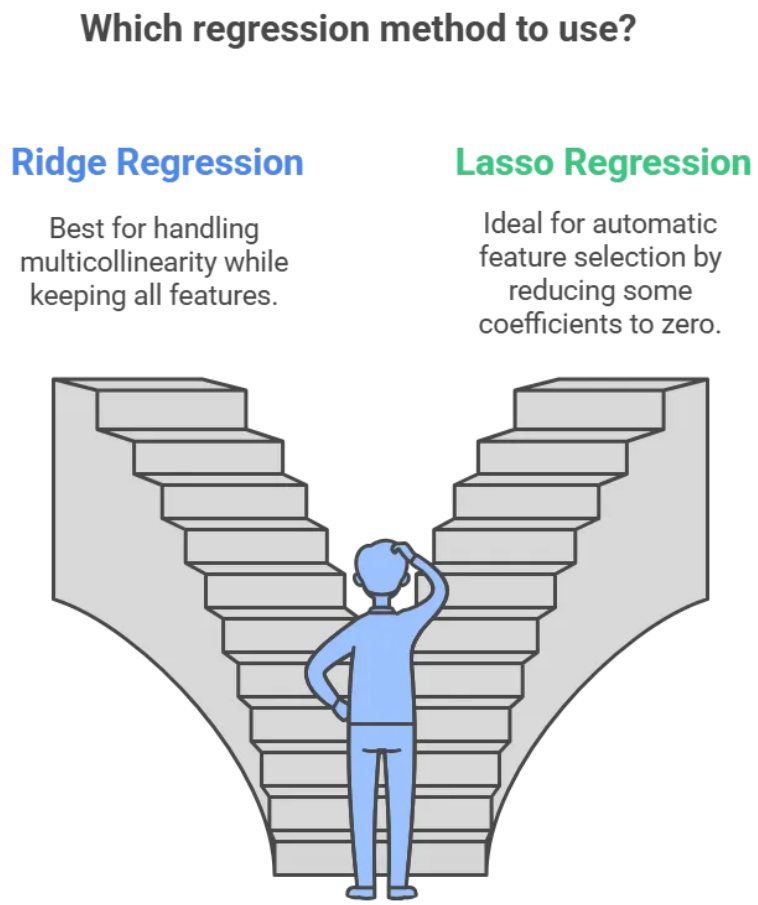
Effects of L1 and L2 regularization in the coefficients of the model

Model: $\hat{y} = 22x_1 - 103x_2 - 14x_3 + 109x_4 - 93x_5 + 203x_6 + 87x_7 - 55x_8 + 378x_9 - 25x_{10} + 8$

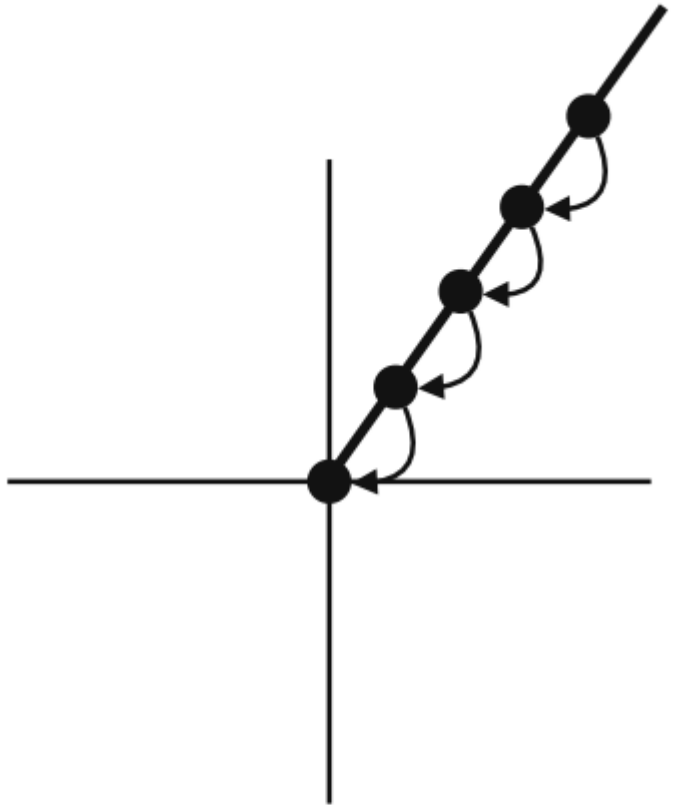
If we add regularization and train the model again, we end up with a simpler model. The following two properties can be shown mathematically:

- If we use L1 regularization (lasso regression), you end up with a model with fewer coefficients. In other words, L1 regularization turns some of the coefficients into zero. Thus, we may end up with an equation like $\hat{y} = 2x_3 + 1.4x_7 - 0.5x_9 + 8$.
- If we use L2 regularization (ridge regression), we end up with a model with smaller coefficients. In other words, L2 regularization shrinks all the coefficients but rarely turns them into zero. Thus, we may end up with an equation like $\hat{y} = 0.2x_1 - 0.8x_2 - 1.1x_3 + 2.4x_4 - 0.03x_5 + 1.02x_6 + 3.1x_7 - 2x_8 + 2.9x_9 - 0.04x_{10} + 8$.

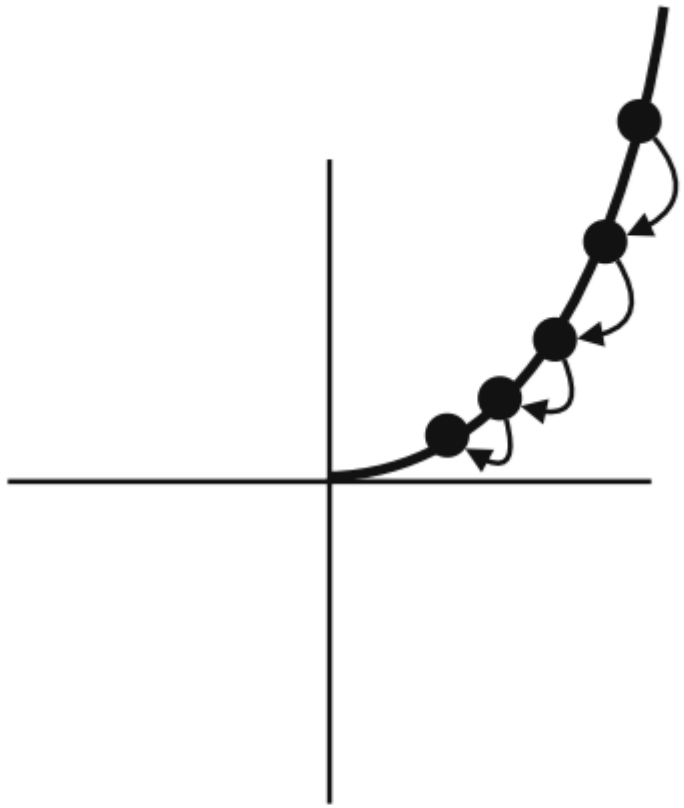
L1 vs. L2



- A quick rule of thumb to use when deciding if we want to use L1 or L2 regularization follows:
  - If we have too many features and we'd like to get rid of most of them, L1 regularization is perfect for that.
  - If we have only few features and believe they are all relevant, then L2 regularization is what we need, because it won't get rid of our useful features.



L1 regularization



L2 regularization

Both L1 and L2 shrink the size of the coefficient. L1 regularization (left) does it much faster, because it subtracts a fixed amount, so it is likely to eventually become zero. L2 regularization takes much longer, because it multiplies the coefficient by a small factor, so it never reaches zero.

Pseudocode for the linear regression algorithm

Inputs: A dataset of points

Outputs: A linear regression model that fits that dataset

Procedure:

- Pick a model with random weights and a random bias.
- Repeat many times:
  - Pick a random data point.
  - Slightly adjust the weights and bias to improve the prediction for that particular data point.
  - **Slightly shrink the coefficients using method 1 or method 2.**
- Enjoy your model!

Regularization Techniques

Exercises

Exercise 4.1

We have trained four models in the same dataset with different hyperparameters. In the following table we have recorded the training and testing errors for each of the models.

Model	Training error	Testing error
1	0.1	1.8
2	0.4	1.2
3	0.6	0.8
4	1.9	2.3

a. Which model would you select for this dataset?

b. Which model looks like it's underfitting the data?

c. Which model looks like it's overfitting the data?

Exercise 4.2

We are given the following dataset:

x	y
1	2
2	2.5
3	6
4	14.5
5	34

We train the polynomial regression model that predicts the value of y as $\hat{y}$, where

$$\hat{y} = 2x^2 - 5x + 4.$$

If the regularization parameter is $\lambda = 0.1$ and the error function we've used to train this dataset is the mean absolute value (MAE), determine the following:

a. The lasso regression error of our model (using the L1 norm)

b. The ridge regression error of our model (using the L2 norm)