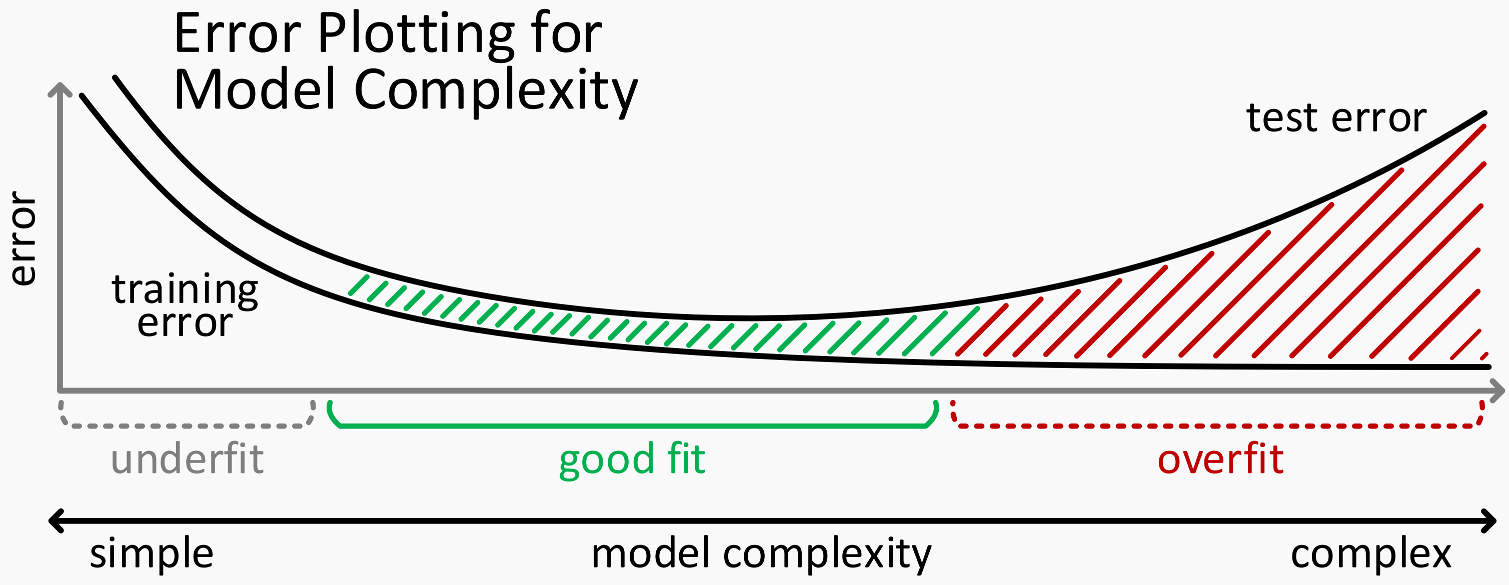
regularization 📊 for overfitting algorithms

Regularization is one of the most significant reasons machine learning models can **generalize**.

Statistical Learning Theory dictates that in order to keep the error small on a test dataset, the model needs to be accurate on the training set while maintaining a certain degree of simplicity:



Allowing a model to become **overly complex** to minimize the **training error** leads to **overfitting**.

Regularization in turn, prevents **overfitting** of the training set through limiting model **complexity**.

Simplicity of a given model is measured by the **Regularization Term**; with significant constant $\boldsymbol{C}$.

$$\frac{1}{n}\sum_{i=1}^{n} \ell\big(y_i f(x_i)\big) + \boldsymbol{C} \times \text{Regulatization}(f)$$

The **Regularization Constant**; constant $\boldsymbol{C}$ is the determinate in balancing accuracy and simplicity.

$$\min_{\text{models } f} \frac{1}{n}\sum_{i=1}^{n} \ell\big(y_i f(x_i)\big) + C + \text{Regulatization}(f)$$

Choose a linear model:

$$f(x_i) = \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \cdots + \beta_p x_{ip}$$

To **Regularize** (**simplify**):

$\beta_1, \beta_2, \beta_3, \ldots, \beta_p$ should be small

Regularization(f) $= \beta_1^2 + \beta_2^2 + \beta_3^2 + \cdots + \beta_p^2 = \|\beta\|_2^2$ (referred to as ℓ_2)

$$\text{Regularization}(f) = |\beta_1^2| + |\beta_2^2| + |\beta_3^2| + \cdots + |\beta_p^2| = \|\beta\|_1 \text{ (referred to as } \ell_1)$$

A Single Dimensional Example:

$$\min_{\text{models } f} \frac{1}{n}\sum_{i=1}^{n} \ell(y_i f(x_i)) + C + \text{Regulatization}(f)$$

Choose a linear model:

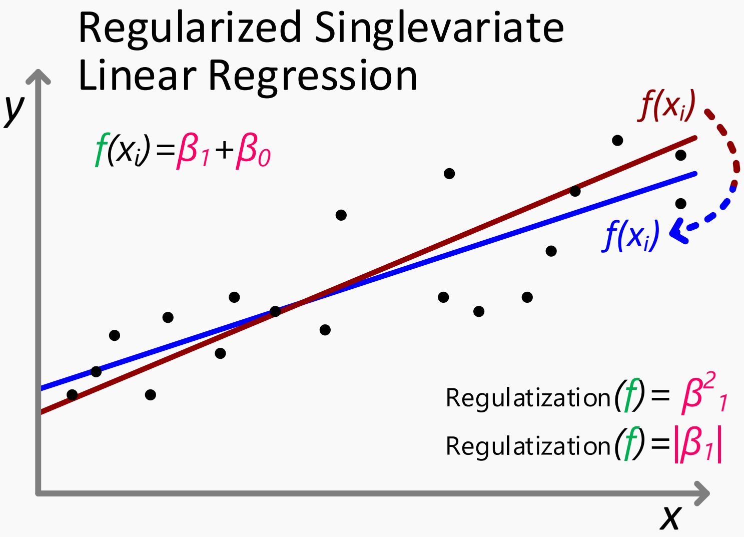
$$f(x_i) = \beta_0 x_i + \beta_1$$

To **Regularize** (**simplify**):

β_1 should be small

$\text{Regularization}(f) = \beta_1^2$ (referred to as ℓ_2)

$\text{Regularization}(f) = |\beta_1|$ (referred to as ℓ_1)


Regularized Singlevariate Linear Regression

Regularization attempts to flatten the linear model as much as it acceptably can. The intent is to avoid influence by excess variance in the dataset. The **Regularization intuition** is more evident if a higher dimensional polynomial term is used in place of a single variable linear expression.

A Single Dimensional Example with a Polynomial Expression:

Choose a linear (polynomial) model:

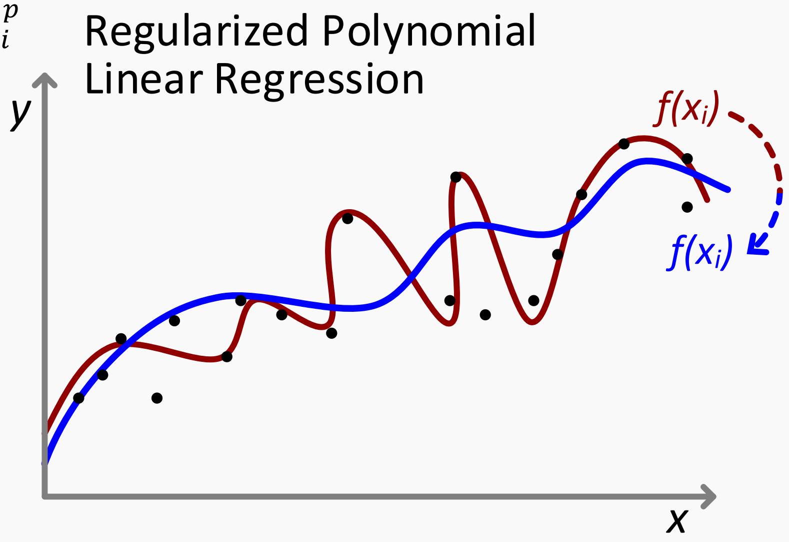
$$f(x_i) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_i^2 + \beta_3 x_i^3 + \cdots + \beta_p x_i^p$$

To **Regularize** (**simplify**):

$\beta_1, \beta_2, \beta_3, \ldots, \beta_p$ should be small

$\text{Regularization}(f) = \beta_1^2 + \beta_2^2 + \beta_3^2 + \cdots + \beta_p^2 = \|\beta\|_2^2$ (referred to as ℓ_2)

$\text{Regularization}(f) = |\beta_1^2| + |\beta_2^2| + |\beta_3^2| + \cdots + |\beta_p^2| = \|\beta\|_1$ (referred to as ℓ_1)


Regularized Polynomial Linear Regression

The Difference Between ℓ_1 and ℓ_2 Regularization

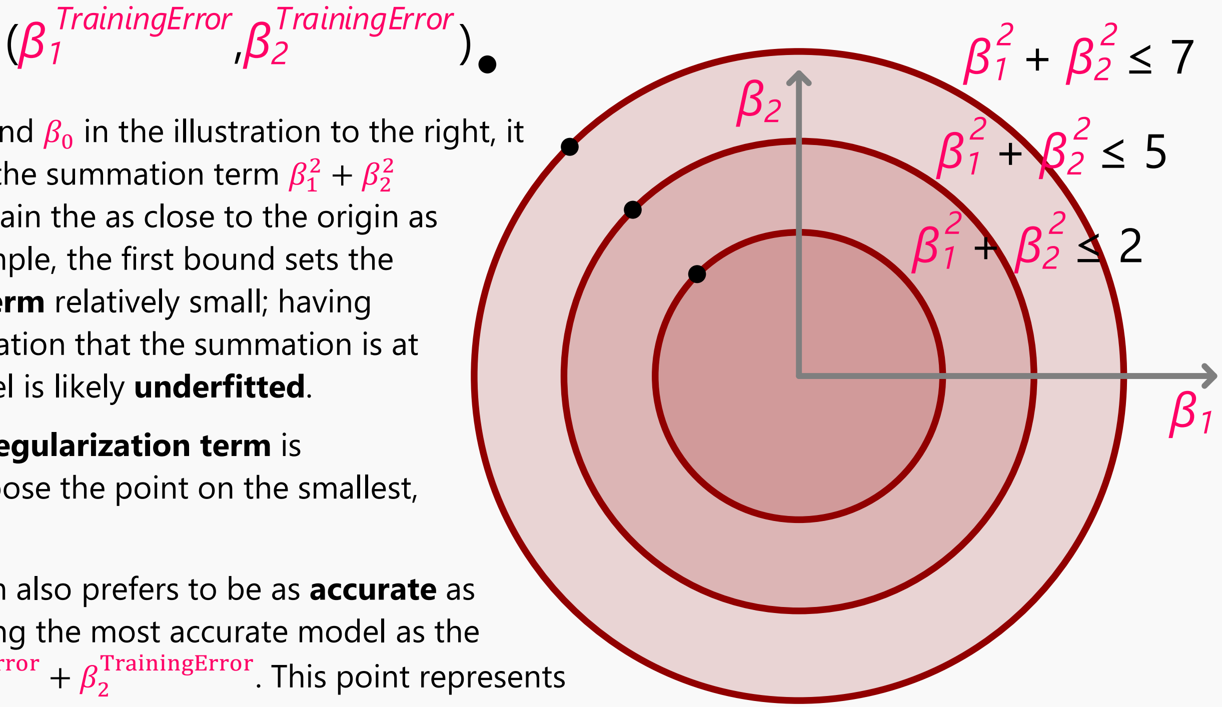
$$\text{Regularization}(f) = \beta_1^2 + \beta_2^2 + \beta_3^2 + \cdots + \beta_p^2 = \|\beta\|_2^2 \text{ (referred to as } \ell_2)$$

ℓ_2 Regularization intuitively tends to make all coefficients slightly smaller.

$$\text{Regularization}(f) = |\beta_1^2| + |\beta_2^2| + |\beta_3^2| + \cdots + |\beta_p^2| = \|\beta\|_1 \text{ (referred to as } \ell_1)$$

ℓ_1 Regularization is particularly useful for making sparse solutions; setting many coefficients $= 0$.

A Geometric 2-Dimensional Example Using $\boldsymbol{\ell_2}$ Regularization:


$(\beta_1^{TrainingError}, \beta_2^{TrainingError})$
$\beta_1^2 + \beta_2^2 \leq 7$
β_2
$\beta_1^2 + \beta_2^2 \leq 5$
$\beta_1^2 + \beta_2^2 \leq 2$
β_1

ℓ_2: Regularization(f) = $\beta_1^2 + \beta_2^2$

Plotting both β_1 and β_0 in the illustration to the right, it can be seen how the summation term $\beta_1^2 + \beta_2^2$ attempts to maintain the as close to the origin as possible. For example, the first bound sets the **Regularization term** relatively small; having enough Regularization that the summation is at most **5**. This model is likely **underfitted**.

In this case, the **Regularization term** is attempting to choose the point on the smallest, inner most circle.

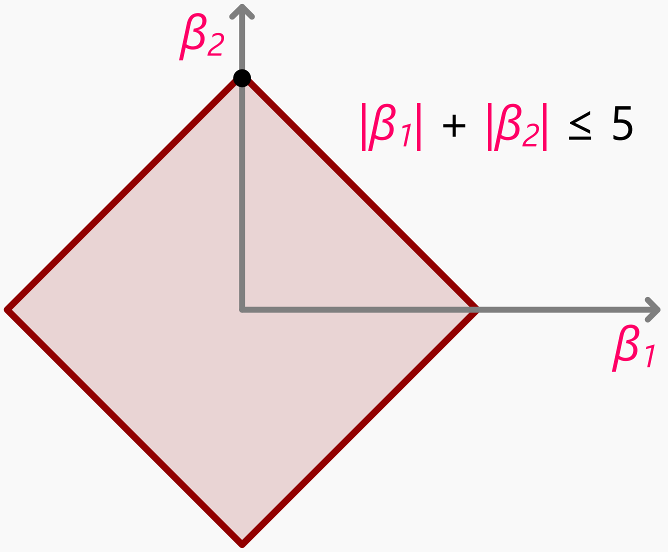
However, the term also prefers to be as **accurate** as possible; illustrating the most accurate model as the point at $\beta_1^{\text{TrainingError}} + \beta_2^{\text{TrainingError}}$. This point represents the model that is the most **overfitted** because it only succeeds at minimizing the **Training Error**. Therefore, the circular gradients ultimately represents regularization options to choose a term that results in a balance between accuracy and simplicity when fitting a model to dataset.

A Geometric 2-Dimensional Example Using $\boldsymbol{\ell_1}$ Regularization:

$$(\beta_1^{TrainingError}, \beta_2^{TrainingError}) \bullet$$

In regards to $\boldsymbol{\ell_1}$ **Regularization**, the gradient actually produces a diamond shape as seen in the illustration to the right. This is due to the **Regularization term** consisting of the absolute values of the term $|\beta_1^2| + |\beta_2^2|$. The term that produces the optimal training error is often one of the terms at the outermost points of the diamond. Specifically illustrated in the model is the optimal point being that of term that sets $\beta_1 = 0$.

This is the intuition of $\boldsymbol{\ell_1}$ **Regularization**. Specifically, in higher dimension spaces, $\boldsymbol{\ell_1}$ **Regularization** tends to prefer sets of coefficients that have many set to 0; the resulting model becomes consequentially sparse with many irrelevant features for making predictions.



ℓ_1: Regularization(f) = $|\beta_1|$ + $|\beta_2|$

Setting up a problem using a linear model and $\boldsymbol{\ell_1}$ **Regularization**:

$$\min_{\text{models } f} \frac{1}{n}\sum_{i=1}^{n} \ell\big(y_i f(x_i)\big) + C + \text{Regulatization}(f)$$

where $\quad f(x_i) = \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \cdots + \beta_p x_{ip}$

$$\begin{aligned} \beta_1 &= 0.7 \\ \beta_2 &= 0 \\ \beta_3 &= 0 \\ \beta_4 &= 0 \\ \beta_5 &= 0 \\ \beta_6 &= 5.6 \\ &\vdots \\ \beta_p &= 0 \end{aligned}$$

The solution to the problem exhibits many of the betas being set = 0; the features are automatically selected, leaving the user to consider only β_1, β_1 and whichever other betas are $\neq 0$.

Regularization(f) $= \beta_1^2 + \beta_2^2 + \beta_3^2 + \cdots + \beta_p^2 = \|\beta\|_2^2$ (referred to as ℓ_2)

$\boldsymbol{\ell_2}$ **Regularization** is also referred to as **Ridge Regression**.

Regularization(f) $= |\beta_1^2| + |\beta_2^2| + |\beta_3^2| + \cdots + |\beta_p^2| = \|\beta\|_1$ (referred to as ℓ_1)

$\boldsymbol{\ell_1}$ **Regularization** is also referred to as the **Lasso Penalty**.

Regularization in general is often referred in whole as **shrinkage**.