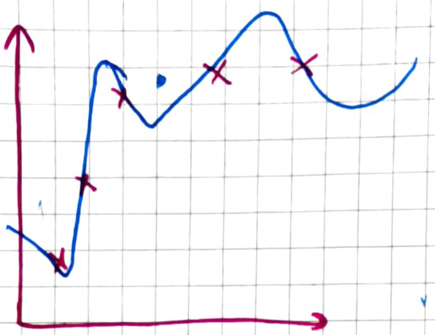
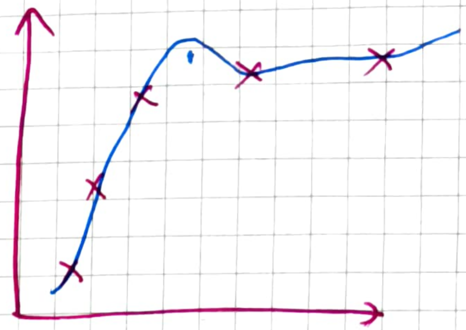
Regularization: Gentle way to reduce the Impact of the feature without being harsh i.e. eliminating the feature Outrightly.

↳ Way to Shrink the Value of the parameters without necessarily demanding that parameters may be set to 0 exactly

Even if you fit a higher Order polynomial using the Small Value of Parameters will end up in a curve which is Still better





$f(x) = 28x - 385x^2 + 39x^3 - 174x^4 + 100$

$f(x) = 13x - 0.23x^2 + 0.000014x^3 - 0.0001x^4 + 10$

Regularization: Keeps all features but prevents the features from having an Overall large effect (which may Cause Overfitting)

→ Regularized Model is Simpler

→ Reduces Overfitting by penalizing wts & thus minimizing the Complexity of model

→ Reduces generalization error / prediction Error

Shrinking weights of higher degree poly will reduce the Complexity of curve/model

---

Ques What is curse of Dimensionality?

If D = 1 $\quad y = wx + b$

If D = 2 $\quad y = w_1 x_1 x_2 + w_2 x_1^2 + w_3 x_1^2 + b$

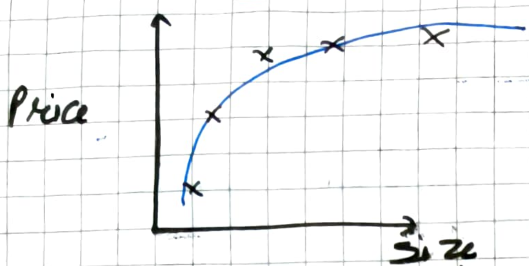
If D = 3 $\quad x_1 x_2 \quad x_1 x_3 \quad x_2 x_3 \quad x_1^2 \quad x_2^2 \quad x_3^2 \quad x_1^2 x_2 \quad x_2^2 x_3 \quad x_3^2 x_1$

As D ↑, no. of Independent Coeff ↑, thus to Capture Complex dependencies in data, we may need to use higher Order polyⁿ.

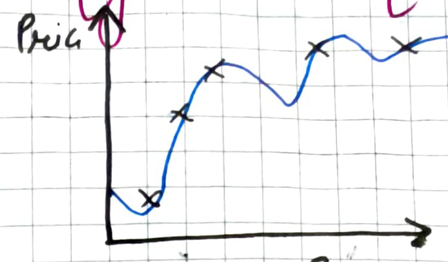
* L1 Reg → used for feature Selection → majority of I/P features will have 0 wt

Cost Function with Regularization

* L2 reg: forces (wts) to be small but doesn't necessarily make them zero
(not robust to outliers)
Soln: Elastic Net L1 + L2



$w_1 x + w_2 x^2 + b$



$w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4 + b$

Make w_3 & w_4 really small (≈ 0)

Penalty Term
Regularization parameter (How much to penalize the wts)

$$\min \frac{1}{2N} \sum \left(y^{(i)} - f(x^{(i)}) \right)^2 + 1000 w_3^2 + 1000 w_4^2$$

→ any large no. will be ok

modified cost

⇒ The func. will penalize the model if w_3 & w_4 are large → Penalizes Higher Values/terms

∵ If we want to min. this cost func., the only way to make it small is to set w_3 & w_4 small enough

otherwise 2 terms will be really really big

⇒ w_3 & w_4 will be close to 0 say 0.0001

⇒ nearly have an impact of getting rid of x^3 & x^4

⇒ will end up fitting curve which is equivalent to quadratic func on left

(∵ Left fit is much better fit)

Idea Behind Regularization → If there are smaller values of parameters we will have a simpler model /

→ May be one with fewer features

→ less prone to overfitting

Regularize/Penalize Weights/parameters

In general, when we have d features, we don't know which features to penalize, thus we penalize all w_j parameters

⇒ Will fit a simpler, smoother curve that's less prone to overfitting

Updated Cost func:

L2 regularization

$$J(w,b) = \frac{1}{2N} \sum_{i=1}^{N} \left(y^{(i)} - f(x^{(i)}) \right)^2 + \frac{\lambda}{2N} \sum_{j=1}^{d} w_j^2 + \frac{\lambda}{2N} b^2$$

optional

So that both terms are scaled similarly

* it turns out that scaling both terms similarly will help you to choose better value of λ.

Regularization Parameter

L1 Reg/Lasso: $\lambda \|w\|$ Penalize absolute value of wts

L2 Reg/Ridge Reg

Objective: To min. first term i.e. mean Sq. error

↳ Encourages algo to fit data well by minimizing diff b/w predictions & actual Values

To min. Second term

↳ To Keep parameters w_j small, which will tend to reduce Overfitting

λ: Controls relative imp./relative Trade off / how you balance b/w 2 goals

Ques. Why λ Should not be too Small or too high?

If $\lambda = 0$ ⇒ No regularization

⇒ Fit Overly wiggly, Overly complex curve.

↳ Effect of overfitting data

Less Training Error, Large Generalization / Prediction Error

If $\lambda \to$ Very very large ⇒ Placing heavy weight on regularization term

⇒ Learning algo will choose $w_1, w_2, \ldots\ldots$ to be extremely close to 0

⇒ $f(x^{(i)})$ is basically equal to b

↳ Learning algo will fit a horizontal Straight line & it underfits

Large Training Error & Large Generalization Prediction Error

Choose λ that balances first & second terms of trade off

↳ Min. mse & keep the parameters small

Regularized Linear Regression (Exactly Same for logistic Regr)

$$\min_{w,b} J(w,b) = \min_{w,b} \left[\frac{1}{2N} \sum_i \left(y^{(i)} - f_{wx+b}(x^{(i)}) \right)^2 + \frac{\lambda}{2N} \sum_{j=1}^{d} w_j^2 \right]$$

$$\frac{d}{dw_3}\left[w_1^2 + w_2^2 + \cdots w_d^2\right]$$

Gradient Descent

repeat

$$w_j = w_j - \alpha \frac{\partial J(w,b)}{\partial w_j}$$

$$b = b - \alpha \frac{\partial J(w,b)}{\partial b}$$

$$\frac{\partial J(w,b)}{\partial w_j} = \frac{1}{N} \sum_{i=1}^{N} \left(f(x^{(i)}) - y^{(i)} \right) x_j^{(i)} + \frac{\lambda}{N} w_j$$

$$\frac{\partial J(w,b)}{\partial b} = \frac{1}{N} \sum_{i=1}^{N} \left(f(x^{(i)}) - y^{(i)} \right)$$

remember we don't regularize b

Why regularization has effect of shrinking w_j

Consider 1st & 3rd term

$$= w_j - \frac{\alpha \lambda}{N} w_j \quad \text{- Usual update}$$

$$= w_j \underbrace{\left(1 - \frac{\alpha\lambda}{N}\right)}_{\text{Shrink } w_j} \quad \text{- Usual update}$$

0.01, 10

ADV → Regularized model is simpler since it has less features
→ reduces Overfitting & thus
→ Reduces the generalization error / prediction error

$$\boxed{J = \frac{1}{2N} \sum_{i=1}^{N} \left(y^{(i)} - \hat{y}^{(i)}\right)^2 + \frac{\lambda}{2N} \sum_{j=1}^{d} w_j^2}$$

$\hat{y}^{(i)} = w_1x_1 + w_2x_2 + \dots w_dx_d + b$

Gradient Descent for

Linear Regression	Regularized Linear Regression
repeat until Convergence	
{ $b = b - \alpha \frac{1}{N} \sum_{i=1}^{N} \left(\hat{y}^{(i)} - y^{(i)}\right)$	{ SAME
$w_j = w_j - \alpha \frac{1}{N} \sum_{i=1}^{N} \left(\hat{y}^{(i)} - y^{(i)}\right) x_j^{(i)}$	$w_j = w_j - \alpha \frac{1}{N} \sum_{i=1}^{N} \left(\hat{y}^{(i)} - y^{(i)}\right) x_j^{(i)} + \frac{\lambda}{N} w_j$
}	}

$$\boxed{J = -\frac{1}{N} \sum_{i=1}^{N} \left(y^{(i)} \log\left(f(x^{(i)})\right) + \left(1 - y^{(i)}\right) \log\left(1 - f(x^{(i)})\right)\right) + \frac{\lambda}{2N} \sum_{j=1}^{d} w_j^2}$$

$f(x^{(i)}) = \frac{1}{1 + e^{-(w_1x_1 + w_2x_2 + \dots w_dx_d + b)}}$

Gradient Descent for

Logistic Regression	Regularized Logistic Regr
repeat until Convergence	
{ $b = b - \alpha \frac{1}{N} \sum \left(f(x^{(i)}) - y^{(i)}\right)$	{ SAME
$w_j = w_j - \alpha \frac{1}{N} \sum_{i=1}^{N} \left(f(x^{(i)}) - y^{(i)}\right) x_j^{(i)}$	$w_j = w_j - \alpha \frac{1}{N} \sum_{i=1}^{N} \left(f(x^{(i)}) - y^{(i)}\right) x_j^{(i)} + \frac{\lambda}{N} w_j$
}	}

Orange: Optional

$$T = \frac{\lambda}{2N} \left(w_1^2 + w_2^2 + w_3^3 + \dots w_d^2\right)$$

$$\frac{\partial T}{\partial w_2} = \frac{\lambda}{\cancel{2}N} * \cancel{2} w_2 = \frac{\lambda}{N} * w_2$$