Regularization

Quick Recap:

✓ Linear Regression: $\boldsymbol{y = f(X) + \epsilon}$,

Where the error term ε, is normally distributed with a mean of zero, $\boldsymbol{\epsilon \sim \mathcal{N}(0, \sigma_\epsilon)}$.

$$\boldsymbol{Err(x) = E\left[\left(y - f(X)\right)^2\right] = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}}$$

➢ Irreducible error, is the noise term in the true relationship that cannot fundamentally be reduced by any model.

- Three kinds of error
  - Inherent: unavoidable
  - Bias: due to over-simplifications
  - Variance: due to inability to perfectly estimate parameters from limited data

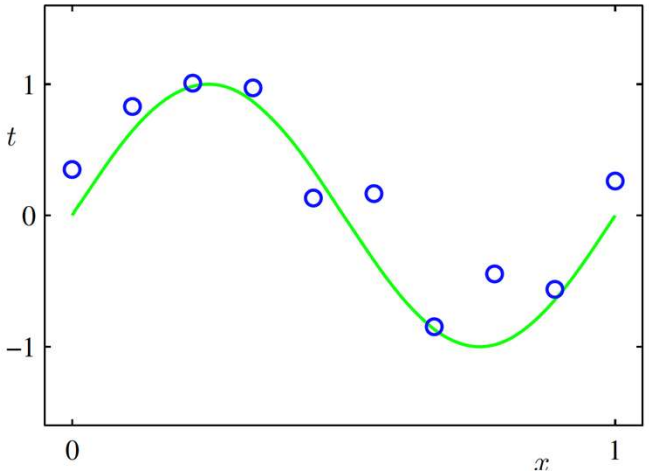
Bias-Variance Trade off

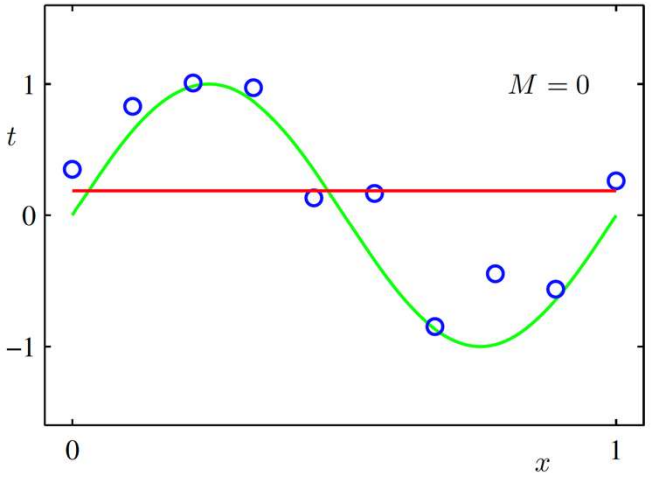
If variance is too high, we have too little data/too complex a function class/etc.
⟹ this is overfitting

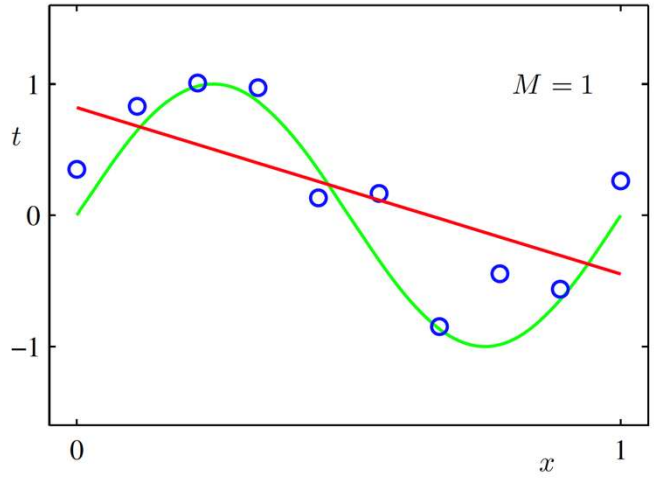
If bias is too high, we have an insufficiently complex function class
⟹ this is underfitting

Remember the bias variance trade-off?

Overfitting/Underfitting

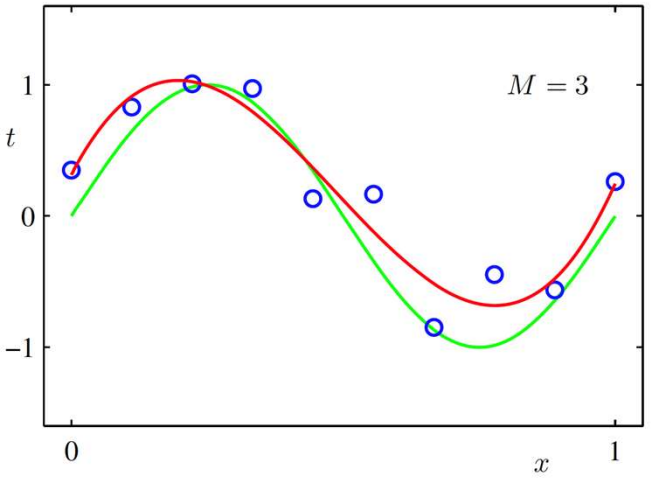


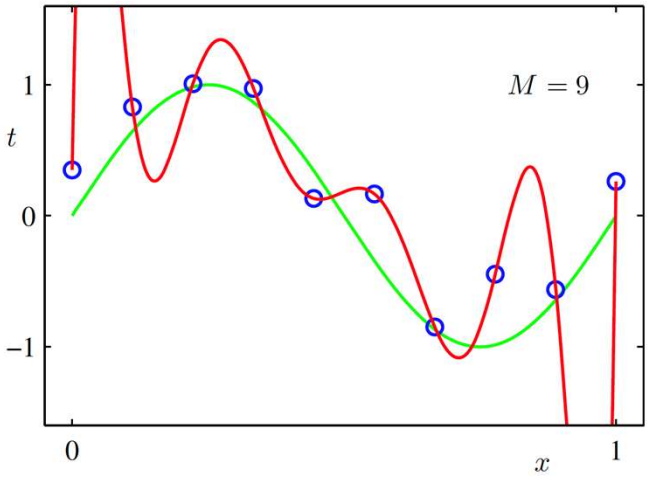




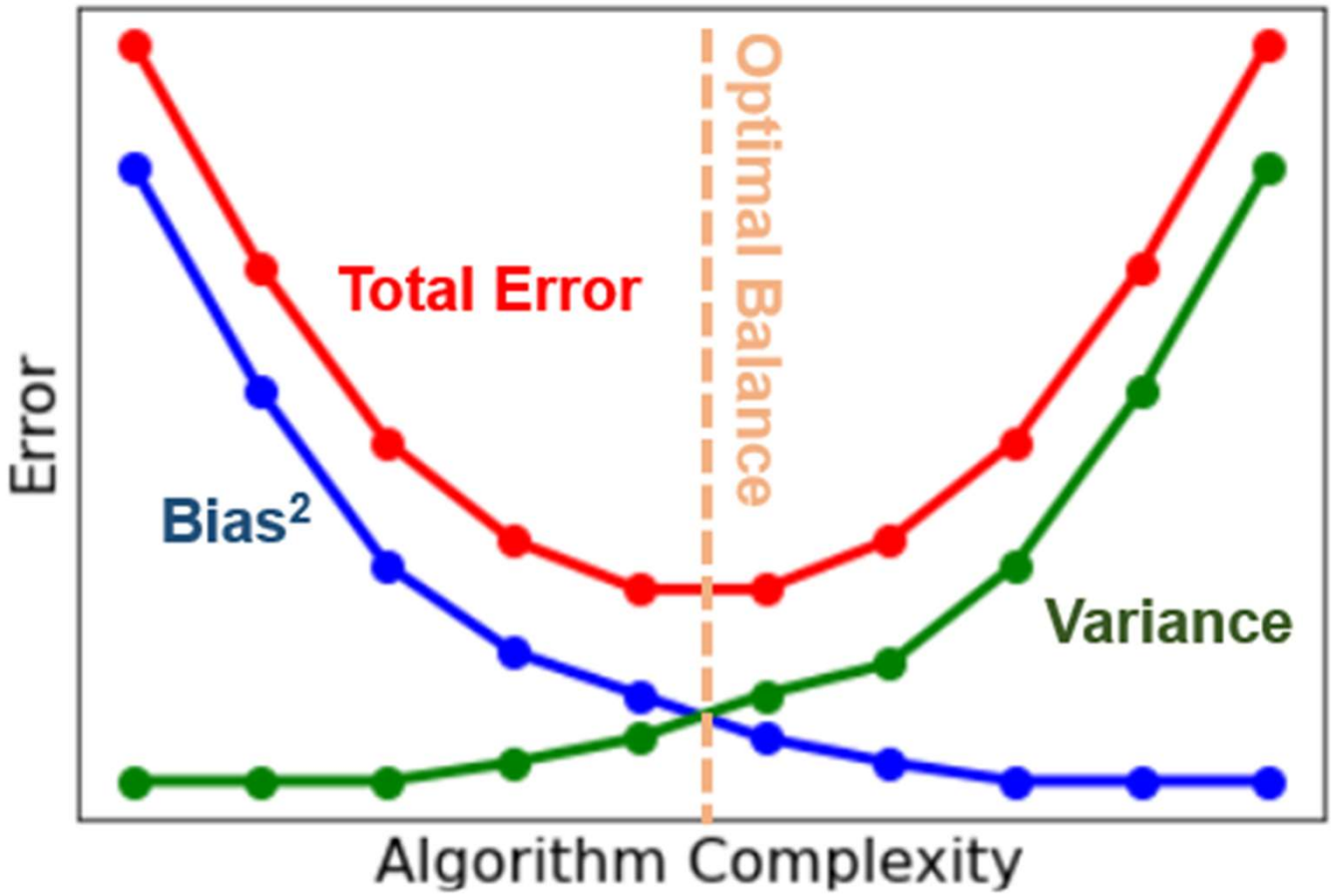
Higher order polynomials will give exact fit on training data reducing the loss function to zero.

We should promote M=3 in this case



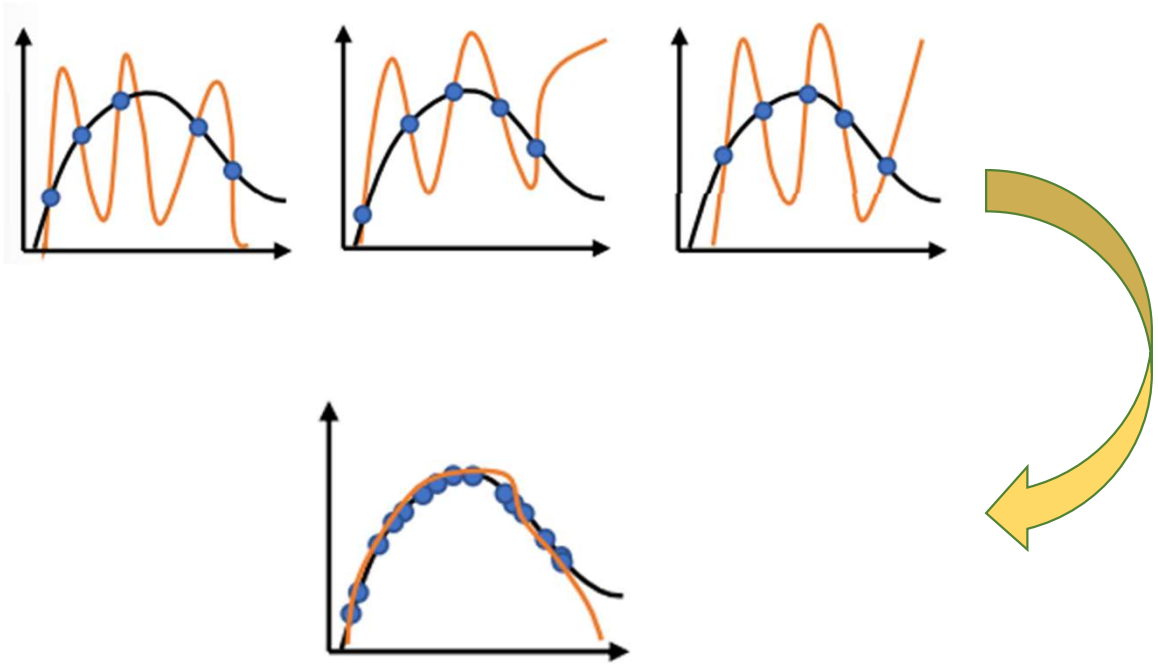


Bias-Variance Trade off



How to reduce variance/Avoid Overfitting?

- ✓Choose a simpler classifier
- ✓Cross-validate the parameters
- ✓Get more training data



Regularization

$$\hat{y} = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + \cdots + w_k x^k$$

- Higher order polynomials gives better fit and lower data loss
- However complicated hypotheses leads to overfitting
- Idea:
  - Change the loss function to penalize hypothesis complexity

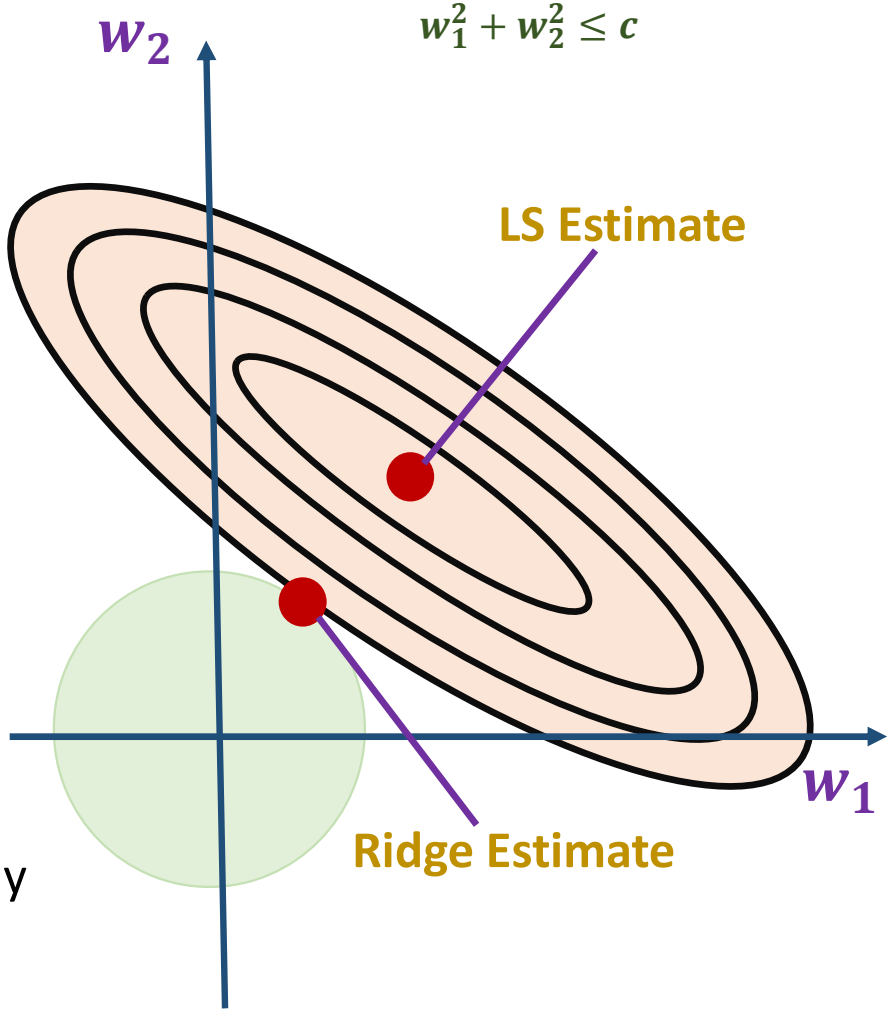
$$\boldsymbol{L(w) = L_D(w) + \lambda\ L_{pen}(w)}$$

- λ is called regularization coefficient and controls how much you value fitting the data well, vs., a simple hypothesis
- This regularization term should promote desired class of solutions

Ridge Regression

$$L_{Ridge} = \frac{1}{2n}\sum_{j=1}^{n}(\hat{y}_j - y_j)^2 + \lambda\frac{1}{2}\sum_{i=1}^{P} w_i^2$$

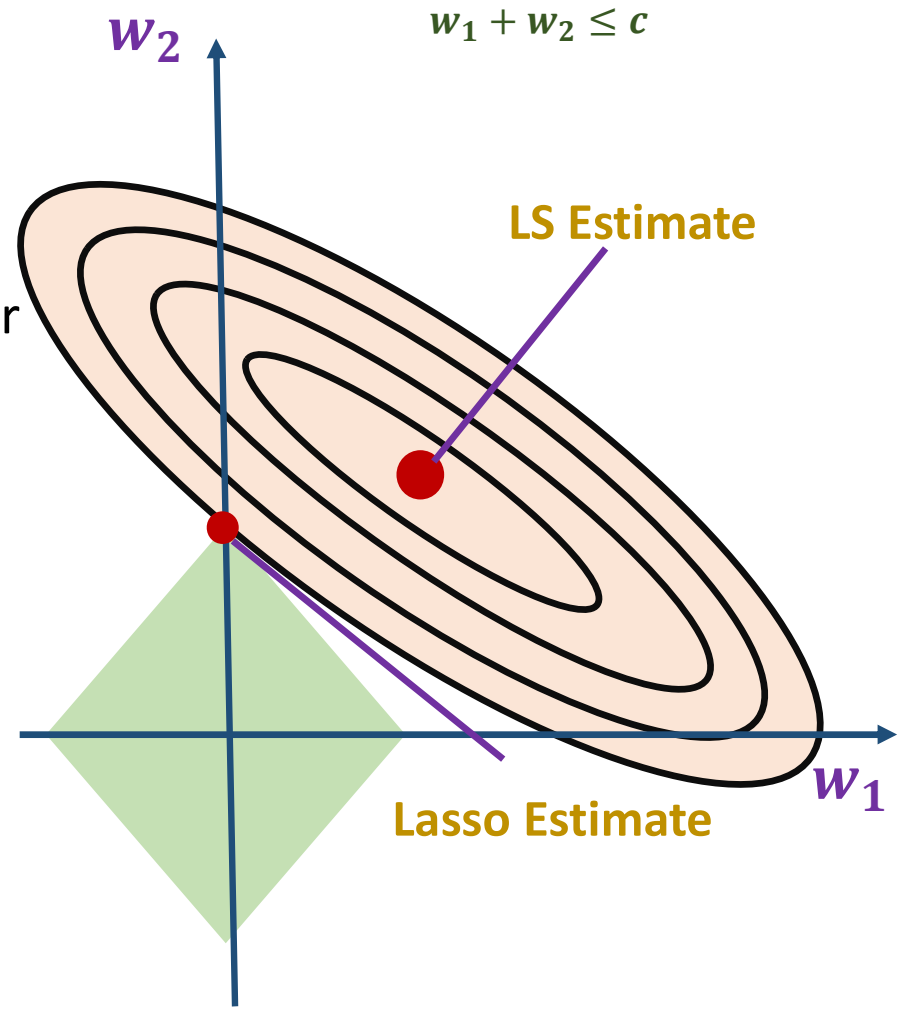
- Also known as L_2 or squared value regularization
- Tries to reduce the length $\|w\|$ of the parameter vector, promoting lesser dependency of $\hat{y}$ on predictors (lower model complexity).
- λ controls the regularization penalty. λ=0 results in regular MSE loss and increase in λ leads to smaller coefficients and lower variance.
- Note that regularizer does not contain w_0 term (only co-efficients of predictors are included).



Lasso Regression

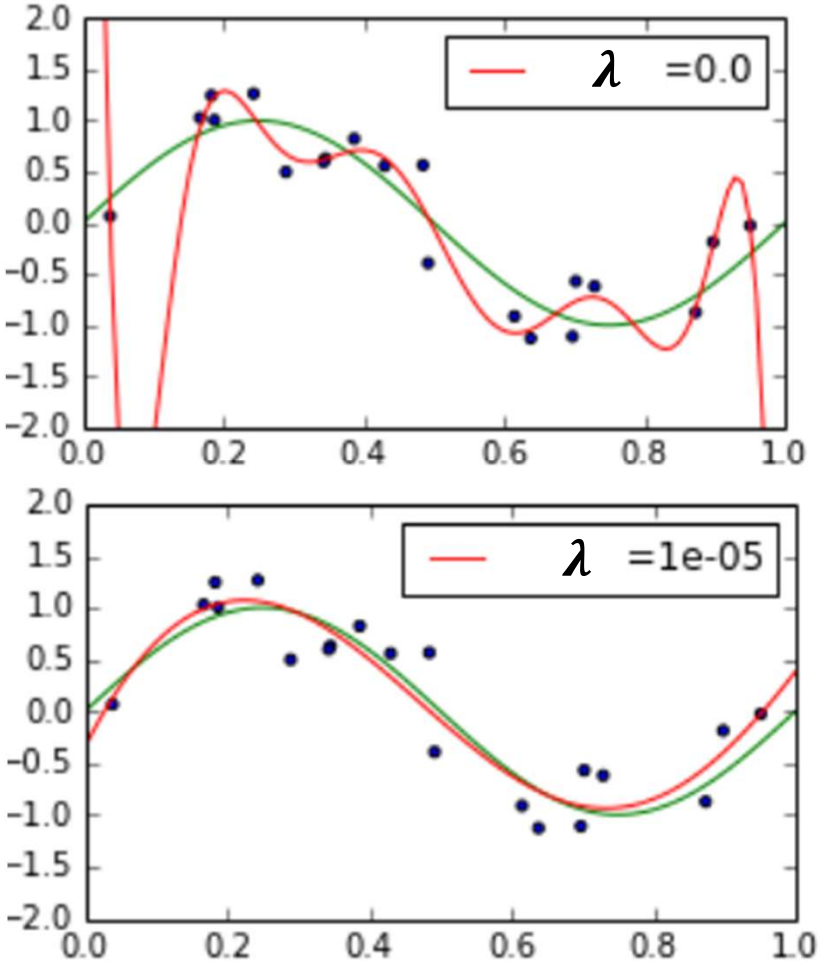
$$L_{Lasso} = \frac{1}{2n}\sum_{j=1}^{n}(\hat{y}_j - y_j)^2 + \lambda \sum_{i=1}^{P}|w_i|$$

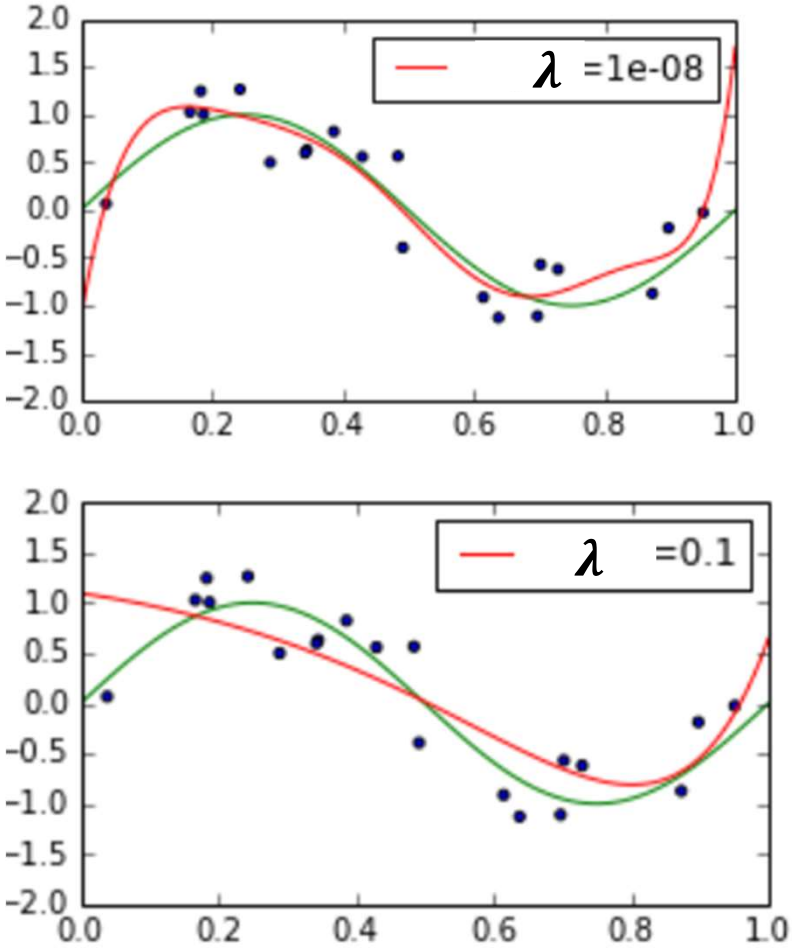
- Least Absolute Shrinkage and Selection Operator (LASSO), also known as L1 or absolute value regularization
- Tries to reduce the city block length $|w|$ of the parameter vector
- λ controls the regularization penalty
- Causes automatic feature selection as useless coefficients are pushed to zero



Regularization Demo

Image Source: Scikit-Learn





Ridge vs. Lasso

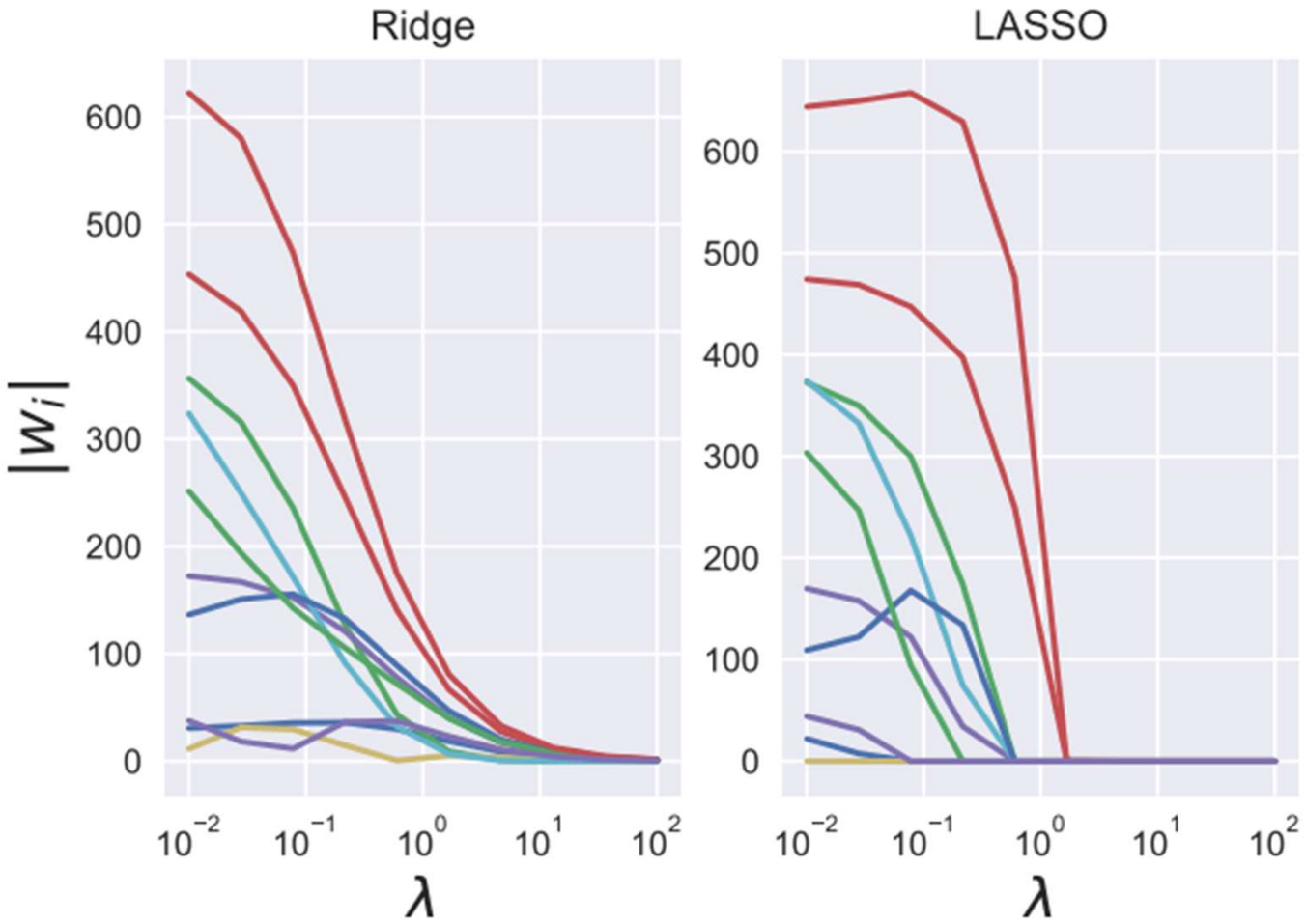
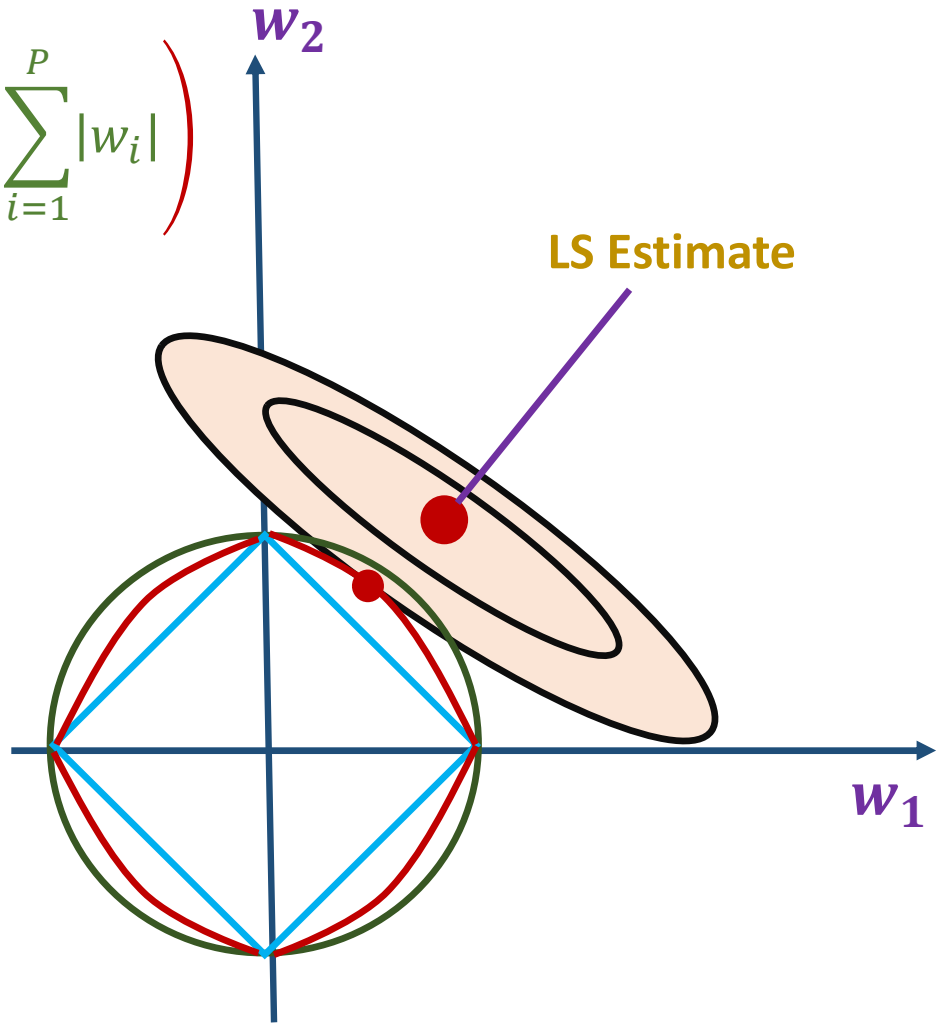


Image Source: https://arxiv.org/pdf/1803.08823.pdf

Elastic Net Regression

$$L_{ElNet} = \frac{1}{2n}\sum_{j=1}^{n}(\hat{y}_j - y_j)^2 + \lambda\left(\frac{(1-\alpha)}{2}\sum_{i=1}^{P} w_i^2 + \alpha\sum_{i=1}^{P}|w_i|\right)$$

- Reduces some of the ill-effects of Lasso Regression and Ridge Regression (see paper below for details).

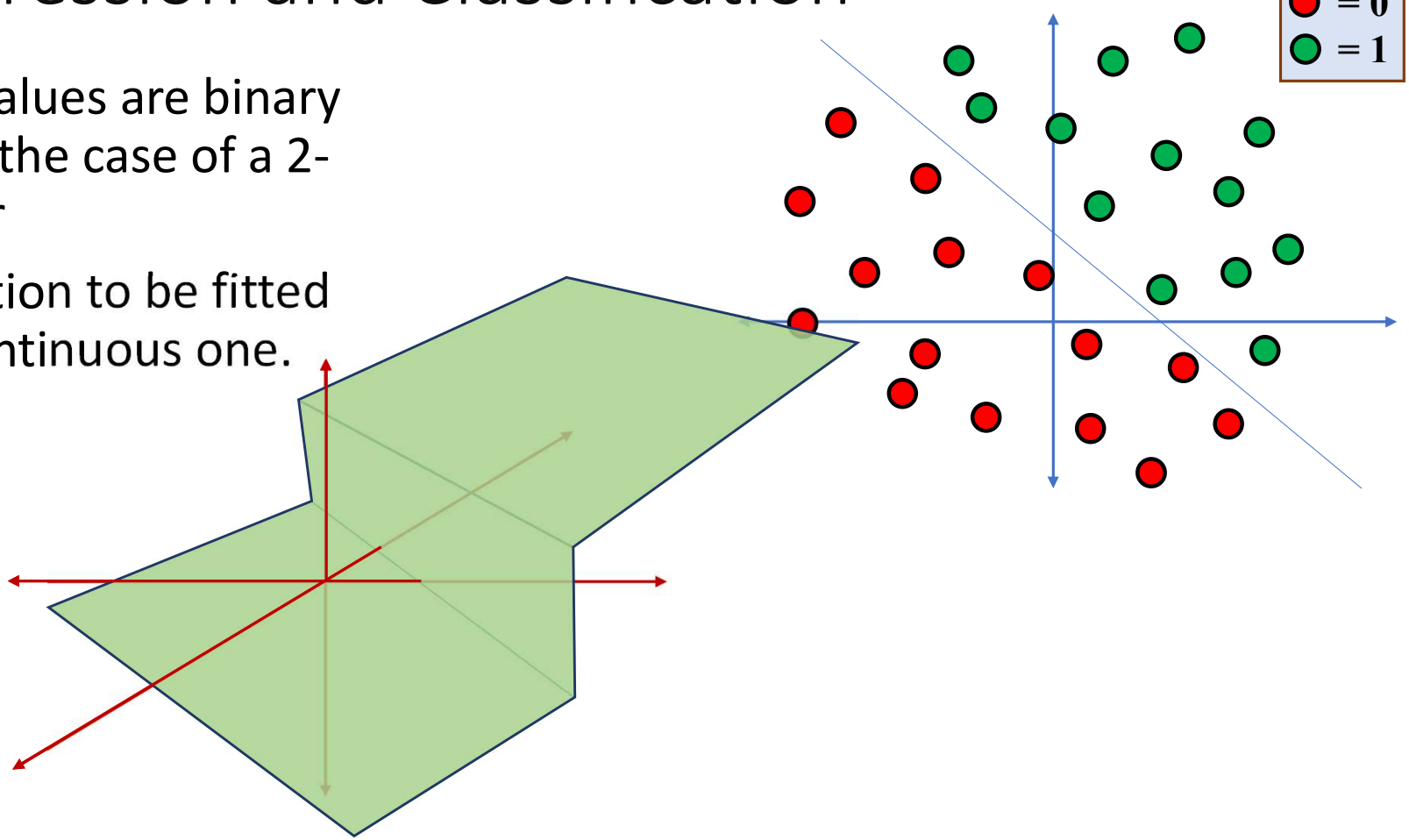


Hui Zou and Trevor Hastie, "Regularization and variable selection via the elastic net", Journal of the Royal Statistical Society, Series B Vol 67(2), 2005, pp. 301–320

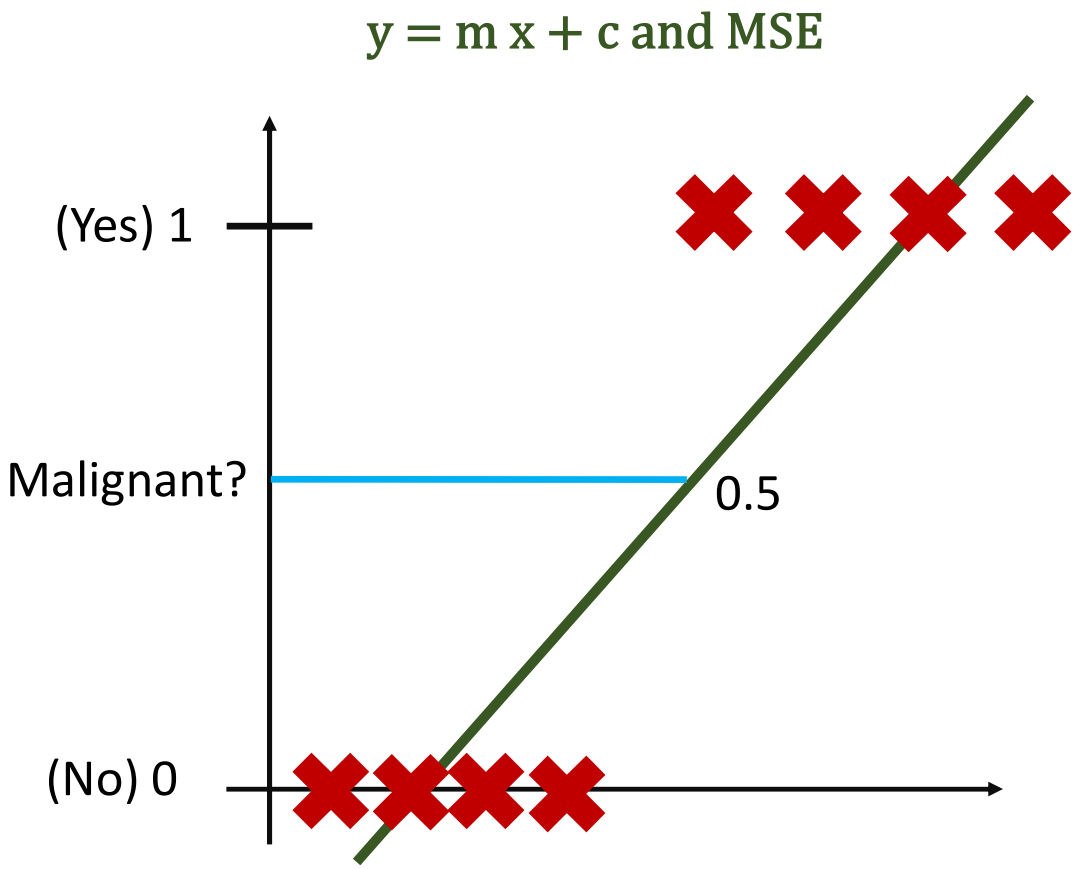
Logistic Regression

Linear Regression and Classification

- Assume the values are binary valued like in the case of a 2-class classifier
- Discrete function to be fitted by a linear continuous one.

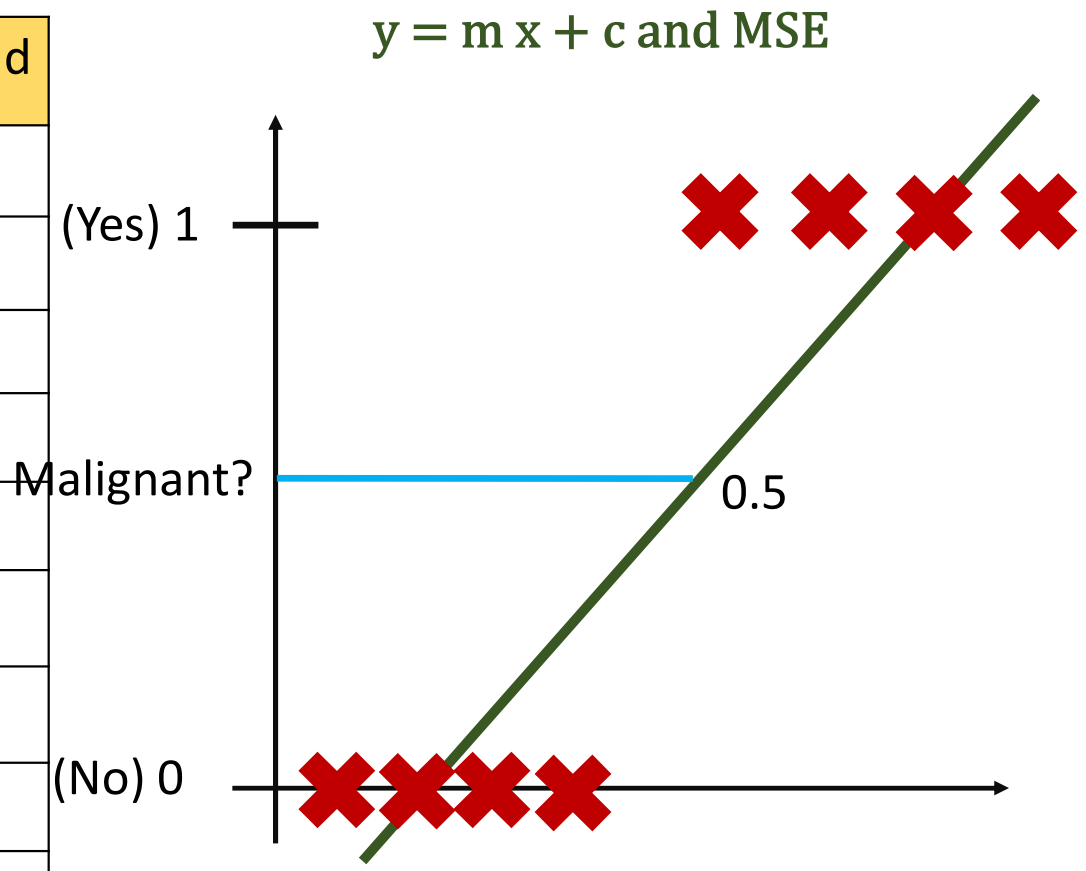


Tumour Size (cm)	Tumour has spread
1.2	0
2.5	0
3.5	0
1.1	0
3.7	1
4.2	1
5.3	1
6.2	1



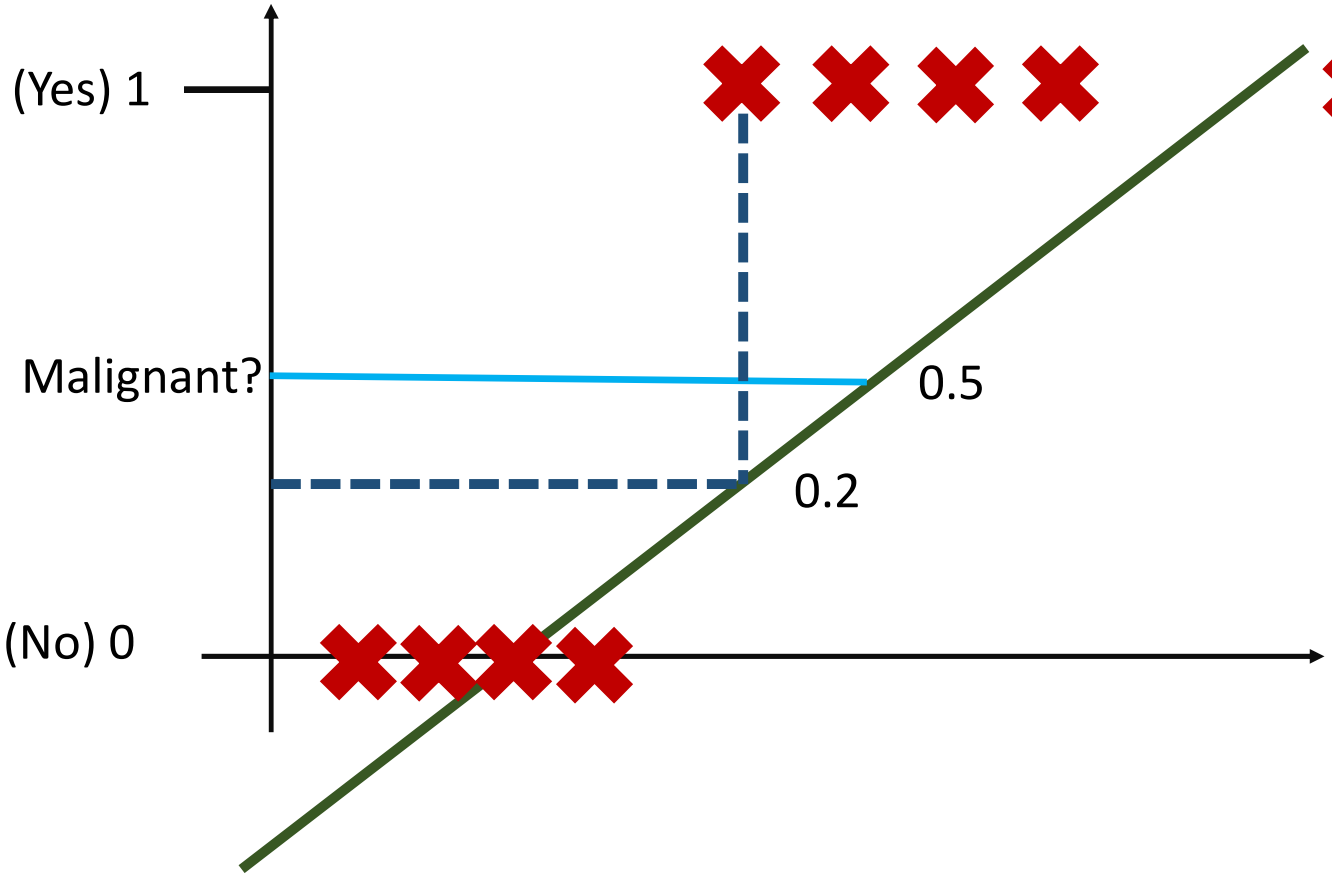
What happens if we have a tumour size of say 11 cm?

Tumour Size (cm)	Tumour has spread
1.2	0
2.5	0
3.5	0
1.1	0
3.7	1
4.2	1
5.3	1
6.2	1
11	1



What happens if we have a tumour size of say 11 cm?

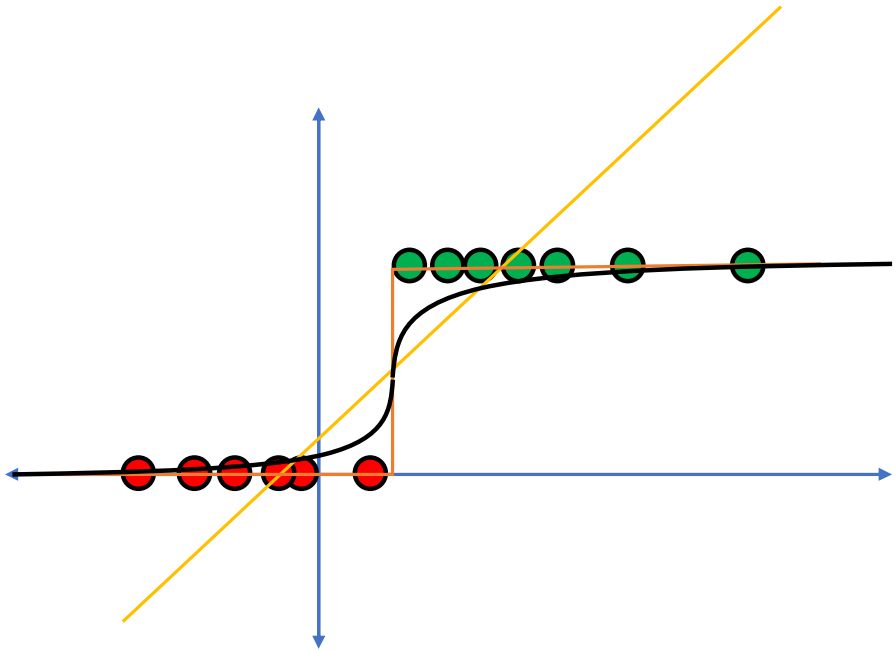
Ans. We run regression line again



- ✓ Linear regression is not the best fit for classification problems!
- ✓ The threshold condition, y(x) > 0.5 (0.2) changes, each time a new sample arrives
- ✓ Look for other alternatives?

Logistic Regression

- Simplified to 1D, the linear approximation clearly works only for a few samples.
- A step function would be good approximation, but difficult to optimize
- Solution: Use a smooth version of step function (logistic fn)



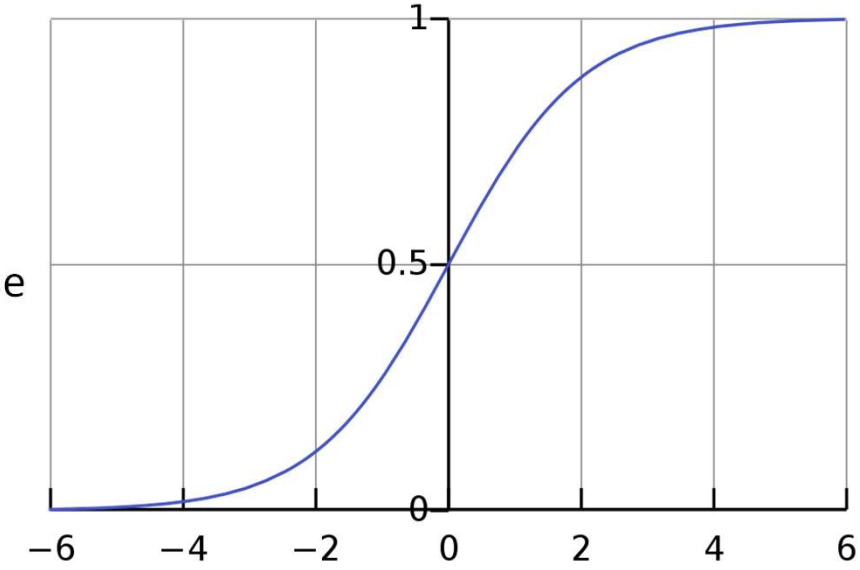
Logistic Regression

Linear Classifier: $z = w_0 + w_1 x_1 + \cdots + w_d x_d$

- The output above is unbounded, whereas it should be between 0 and 1, i.e. $z = [0 \leq p(X) \leq 1]$
- Changing p by the same amount requires a bigger change in x when p is already large (or small) than when p is close to 1/2

$$\log \frac{p(x)}{1 - p(x)} = w_0 + w_1 x_1 + \cdots + w_d x_d$$

$$p(X) = \frac{e^z}{1 + e^z} = \frac{1}{1 + e^{-z}}$$



Sigmoid Function: It has the property of mapping the entire number line into a small range, between 0 and 1

Types of Logistic Regression

- **Binomial:** In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.

- **Multinomial:** In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"

- **Ordinal:** In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Logistic Regression Cost Function

Loss Function

- In binary classification, the cross-entropy loss is defined as:

$$L(y, p) = -(y \log p + (1 - y) \log(1 - p))$$

  - ***It is a measure of the difference between the predicted probability p and the true probability y.***

- In multi-class classification problems, the cross entropy loss is calculated for each class separately and then summed together

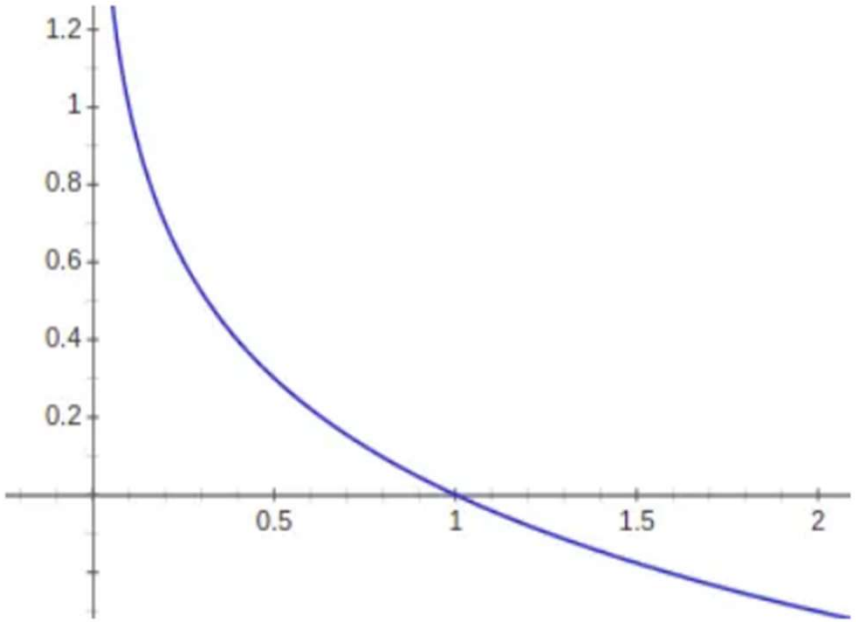
$$L(y, p) = -\sum_{i=1}^{n} y_i \log p_i, \quad \text{for n classes}$$

$$\mathrm{p}(\mathbf{X}) = \frac{1}{1+e^{-(\mathbf{w}^{\mathbf{T}}\mathbf{X})}}$$

Loss Function, y=1

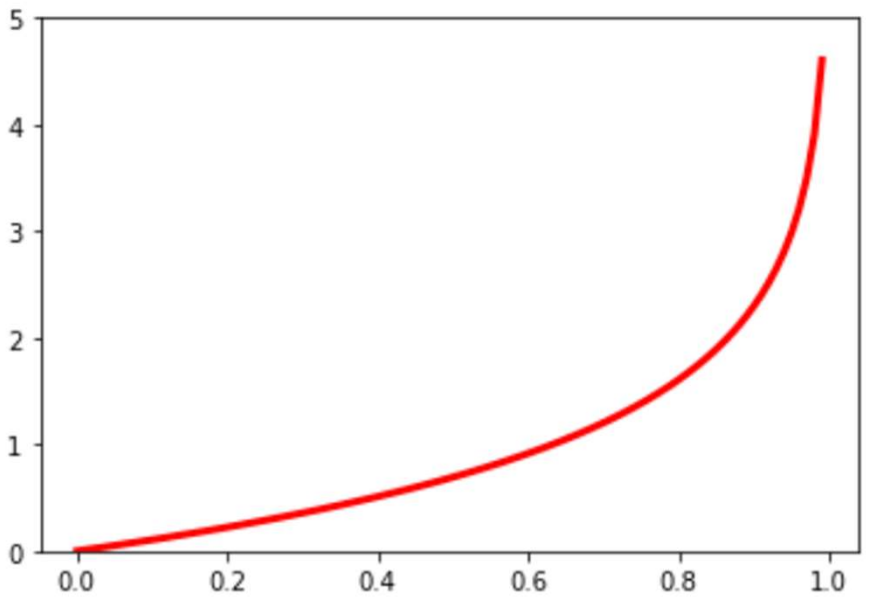
$$L(y, p) = -(y \log p + (1 - y) \log(1 - p))$$
$$= -\log p$$

- ✓ When model's prediction is closer to 1, the penalty is closer to 0 .
- ✓ As the prediction moves further from 1 and towards 0, the penalty increases.



Loss Function, y = 0

$$L(y, p) = -(y \log p + (1 - y) \log(1 - p))$$
$$= -\log(1 - p)$$



- ✓ When model's prediction is closer to 1, the penalty is closer to infinity .
- ✓ As the prediction moves further from 1 and towards 0, the penalty tends to zero.

Gradients of Logistic Regression

- $z = x\,w + b$
- $p(x) = \frac{1}{1+e^{-z(x)}}$

$$\textbf{cost function} = -y\,\log p(x) - (1-y)\,\log\big(1-p(x)\big)$$

$$= -y\,\log\frac{1}{1+e^{-(x\,w+b)}} - (1-y)\log\frac{1}{1+e^{(x\,w+b)}}$$

❑ We need to find $\frac{\partial \text{cost}}{\partial w}$, and $\frac{\partial \text{cost}}{\partial b}$

✓ $\frac{\partial cost}{\partial w} = x(p(x) - y).$

✓ $\frac{\partial cost}{\partial b} = (p(x) - y)$

Summary

- Logistic Regression is used for predicting the categorical dependent variable using a given set of independent variables

- The outcome can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1