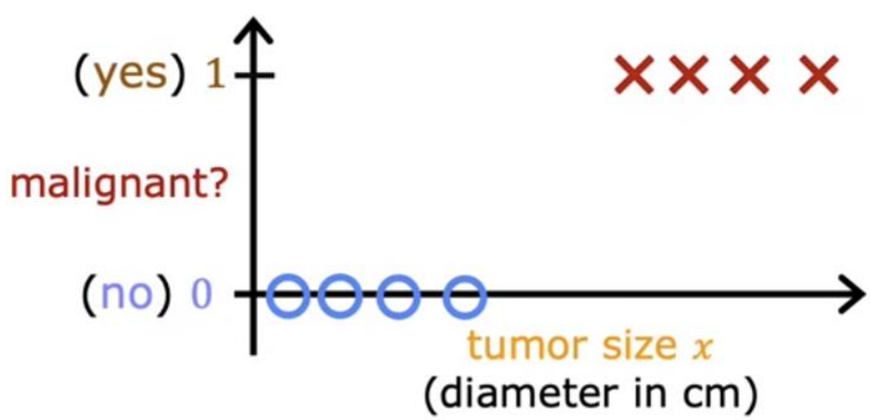
Week 3: Classification

<u>Classification with Logistic Regression</u>

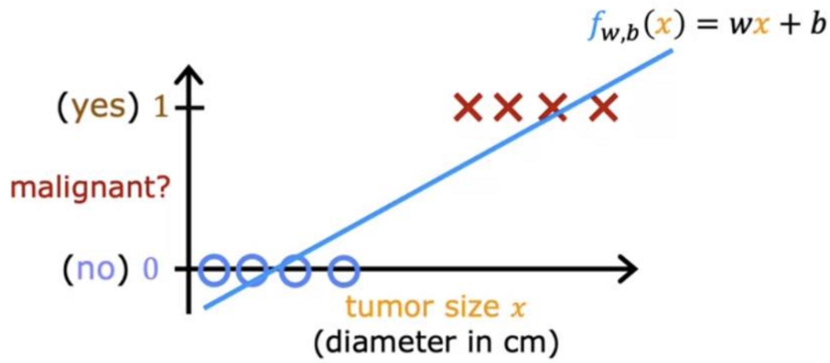
- ➔ Motivations
  - ◆ In classification, y can only be a small range of possible values

Question	Answer "y"	
Is this email spam?	no	yes
Is the transaction fraudulent?	no	yes
Is the tumor malignant?	no	yes

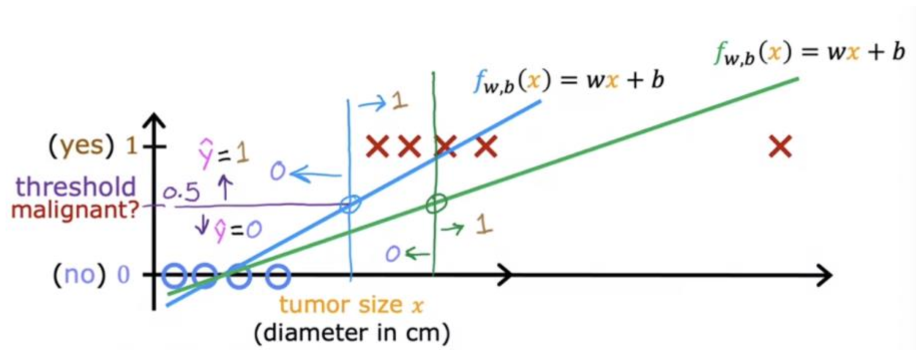
    - In this case, *y* can only be one of two values → **binary classification**
    - Common y values
      - Could be no or yes/false or true/0 or 1 (all of these refer to the same things
        - ◆ 0 is referred to as the negative class, while 1 is referred to as the positive class
  - ◆ In classification, class = category (terminology)
    - Can be used interchangeably
  - ◆ Could you apply linear regression to classification algorithms?
    - Example:



- Could have a best fit line like this:



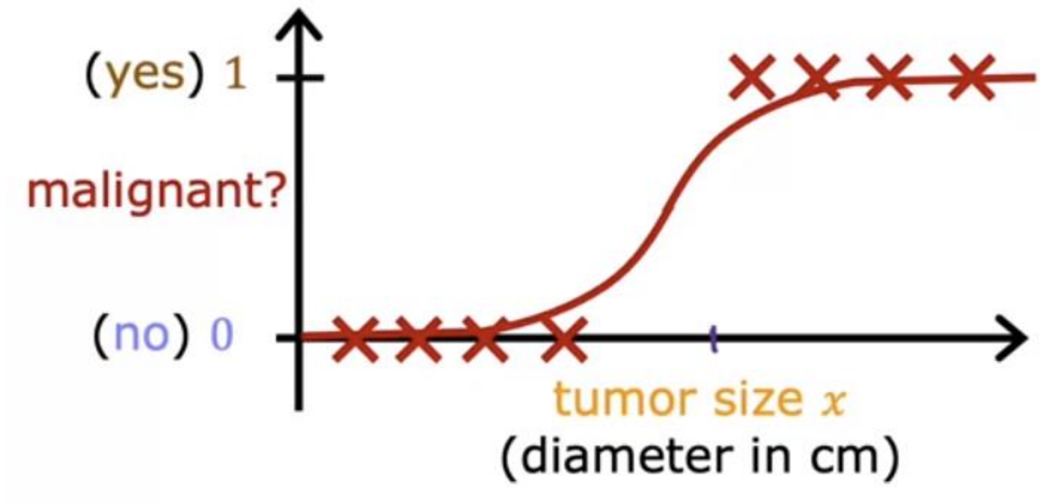
  - Linear regression does not only predict the values of 0 and 1; it predicts all values → our goal though is to predict categories
    - Could use a threshold approach
      - $if\ f_{w,b}(x) < 0.5 \rightarrow \hat{y} = 0\ (not\ malignant)$
      - $if\ f_{w,b}(x) \geq 0.5 \rightarrow \hat{y} = 1\ (malignant)$
- But this doesn't work for all training sets



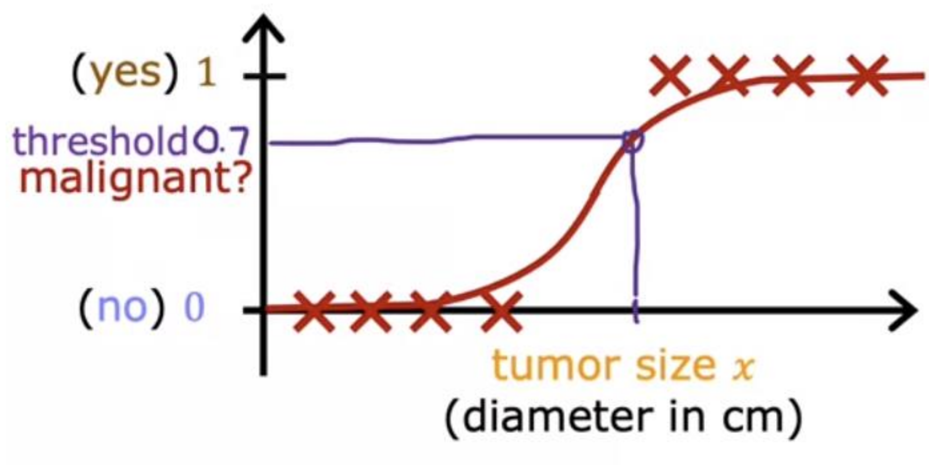
  - If we add one more training example to the right of the graph, then doesn't work anymore as some malignant training examples will be classified as 0 as the output (some values are misclassified)

→ Logistic Regression
  - Classification algorithm

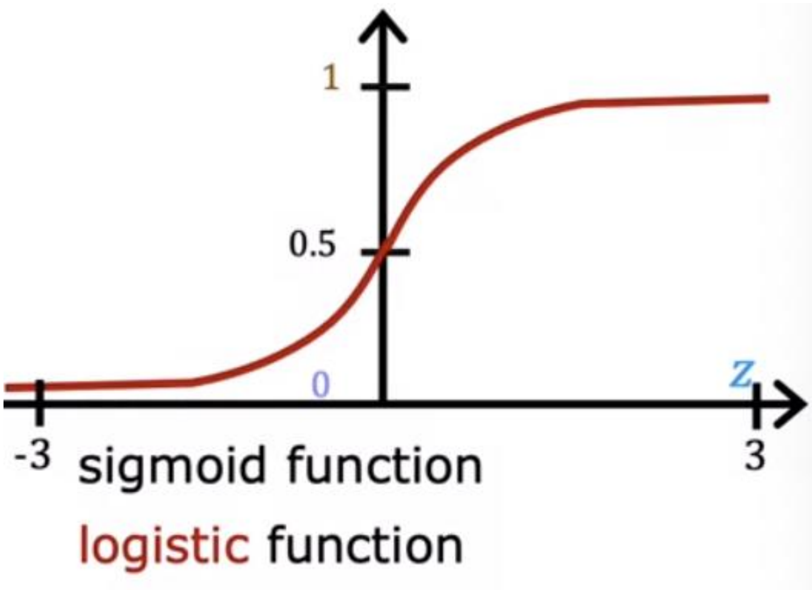
- Example Data Set:



- Fits an S sized curve to the data → different threshold than linear regression



- Sigmoid Function / Logistic Function



  - Outputs values between 0 and 1
    - $g(z) = \frac{1}{1+e^{-z}}$ $where\ 0 < g(z) < 1$
      - When *z* is large → the value of *g(z)* will be close to 1
      - When z is small → the value of *g(z)* will be close to 0
- Logistic Regression Algorithm
  - Start with $z = \vec{w} \cdot \vec{x} + b$
    - → pass it to $g(z) = \frac{1}{1+e^{-z}}$
      - → when combined they give you the logistic regression model $f_{\vec{w},b}(x) = g(\vec{w} \cdot \vec{x} + b) = g(z) = \frac{1}{1+e^{-(\vec{w} \cdot \vec{x} + b)}}$
  - Interpeting the output
    - "Probability" the class is 1 $= P(y = 1|x; \vec{w}, b)$
      - Equation means the probability *y = 1* given *x* and the parameters $\vec{w}\ and\ b$
    - Example:

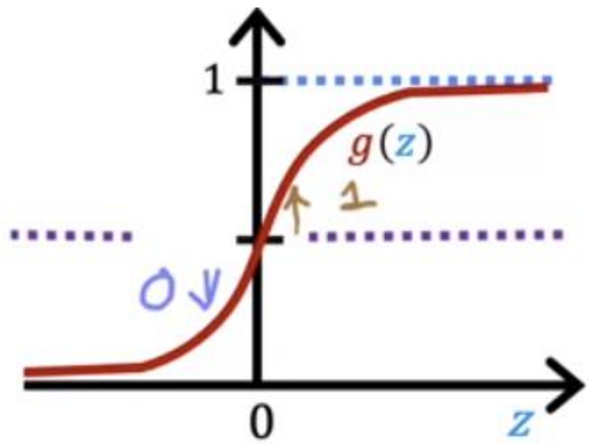
      x is "tumor size"
      y is 0 (not malignant)
      or 1 (malignant)

      $f_{\vec{w},b}(\vec{x}) = 0.7$

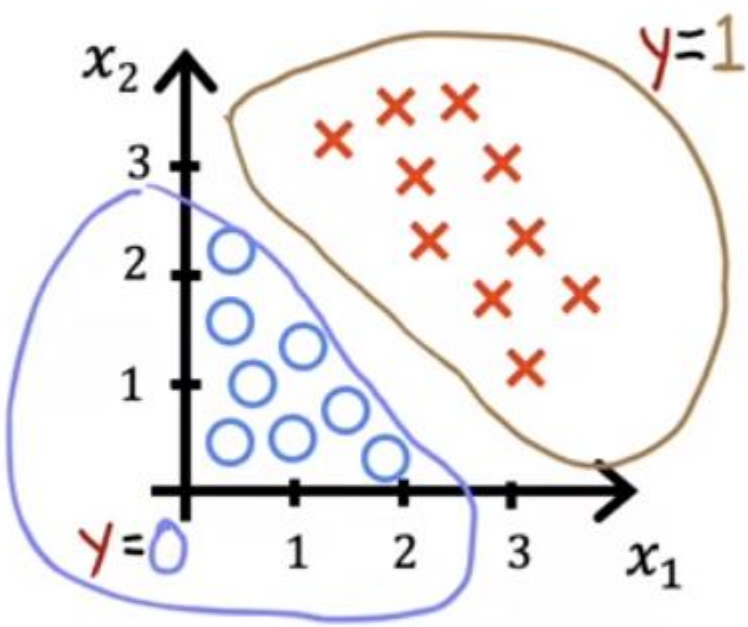
      - Model thinks there is a 70% chance that *y* is 1 for this patient

                - 30% chance that y is 0 for this patient

- Decision Boundary
    - You want the learning algorithm to decide whether to assign the y value 0 or 1



        - Could set threshold
            - Common choice to set threshold = 0.5
                - $if\ f_{w,b}(x)\ <\ 0.5\ \rightarrow\ \hat{y}\ =\ 0$
                    - $\rightarrow =\ \vec{w}\ \cdot\ \vec{x}\ + b\ \ < 0$
                - $if\ f_{w,b}(x)\ \geq\ 0.5\ \rightarrow\ \hat{y}\ =\ 1$
                    - $=\ g(z)\ \geq\ 0.5$
                    - $=\ z\ \geq\ 0$
                    - $=\ \vec{w}\ \cdot\ \vec{x}\ + b\ \geq\ 0$
    - How does the model make a prediction?
        - Example with two features:

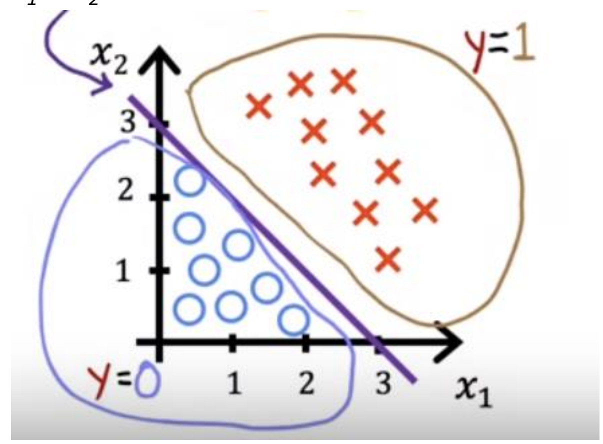


            - $\rightarrow$ Will use this model: $f_{\vec{w},b}(\vec{x})\ =\ g(z)\ =\ g(w_1x_1 + w_2x_2 + b$

- Pretend:

$$f_{\vec{w},b}(\vec{x}) = g(z) = g(\underset{1}{w_1} x_1 + \underset{1}{w_2} x_2 + \underset{-3}{b})$$

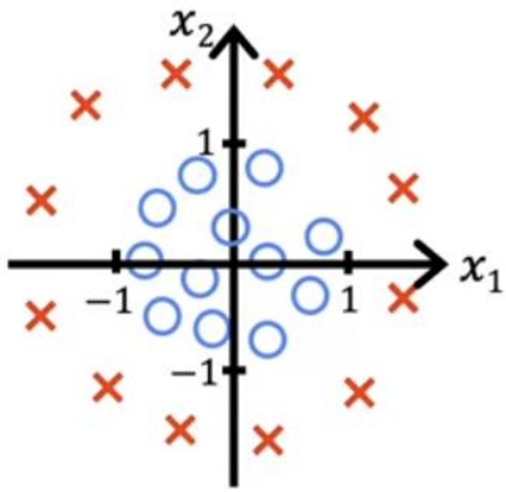
  - $z = x_1 + x_2 - 3$
  - Look at when $z = \vec{w} \cdot \vec{x} + b = 0$
    - This line when z = 0 is referred to as the **decision boundary**
      - Neutral about y being 0 or 1
    - For what x values is $z = x_1 + x_2 - 3 = 0$?
      - $x_1 + x_2 = 3$



        - → Results in the purple line (decision boundary)
        - If to the right of the line → algorithm predicts 1
        - If to the left of the line → algorithm predicts 0

- Non-linear decision boundaries

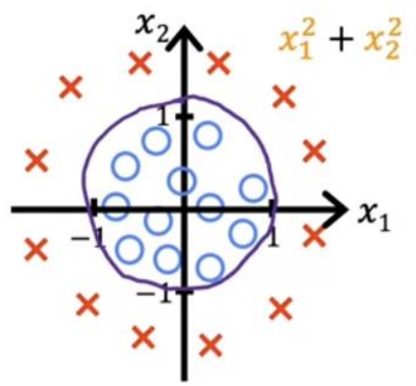
- You could use polynomials to get decision boundaries



  - Set $g(z) = w_1 x_1^2 + w_2 x_2^2 + b$
    - Pretend

$$f_{\vec{w},b}(\vec{x}) = g(z) = g(\overbrace{\underset{1}{w_1} x_1^2 + \underset{1}{w_2} x_2^2 + \underset{-1}{b}}^{z})$$

- $\rightarrow z = x_1^2 + x_2^2 - 1 = 0$ to get decision boundary
  - $\rightarrow x_1^2 + x_2^2 = 1$



- You get the following boundary

- To get more complex boundaries, you can use higher order polynomial terms

Practice Quiz: Classification with Logistic Regression

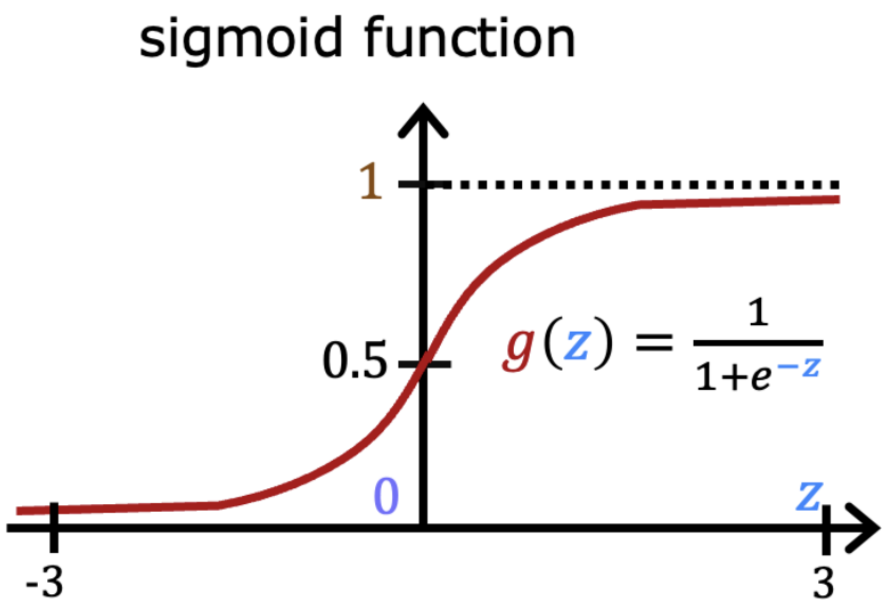
1. Which is an example of a classification task?

1 point

- ○ Based on a patient's blood pressure, determine how much blood pressure medication (a dosage measured in milligrams) the patient should be prescribed.
- ○ Based on a patient's age and blood pressure, determine how much blood pressure medication (measured in milligrams) the patient should be prescribed.
- ◉ Based on the size of each tumor, determine if each tumor is malignant (cancerous) or not.

2. Recall the sigmoid function is $g(z) = \frac{1}{1+e^{-z}}$

1 point

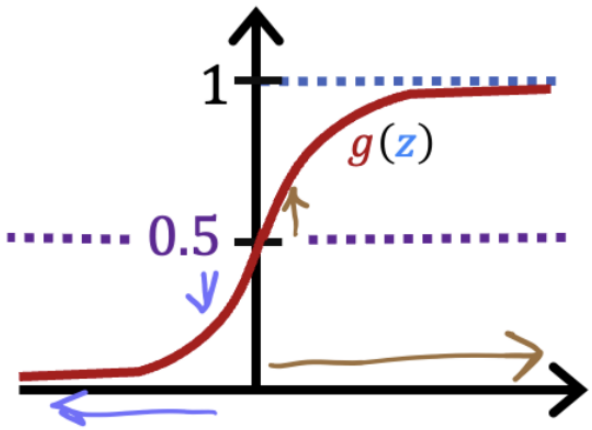


If z is a large positive number, then:

- ○ $g(z)$ will be near 0.5
- ○ $g(z)$ is near negative one (-1)
- ◉ $g(z)$ is near one (1)
- ○ $g(z)$ will be near zero (0)

3.

1 point



A cat photo classification model predicts 1 if it's a cat, and 0 if it's not a cat. For a particular photograph, the logistic regression model outputs $g(z)$ (a number between 0 and 1). Which of these would be a reasonable criteria to decide whether to predict if it's a cat?

- ○ Predict it is a cat if g(z) < 0.5
- ○ Predict it is a cat if g(z) = 0.5
- ◉ Predict it is a cat if g(z) >= 0.5
- ○ Predict it is a cat if g(z) < 0.7

4.

1 point

True/False? No matter what features you use (including if you use polynomial features), the decision boundary learned by logistic regression will be a linear decision boundary.

- ○ True
- ◉ False

Cost Function for Logistic Regression

- Cost Function for Logistic Regression
  - Example training model

	tumor size (cm) x_1	...	patient's age x_n	malignant? y
$i=1$	10		52	1
:	2		73	0
	5		55	0
	12		49	1
$i=m$	...		...	...

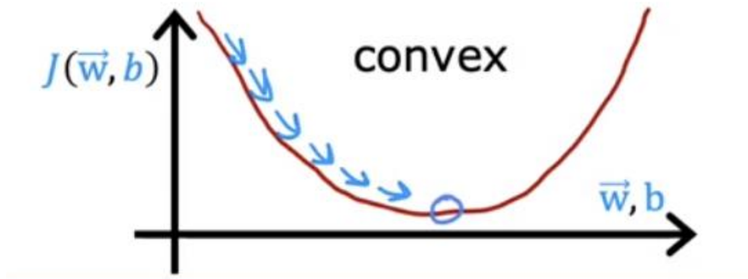
$i = 1, \dots, m \leftarrow$ training examples

$j = 1, \dots, n \leftarrow$ features

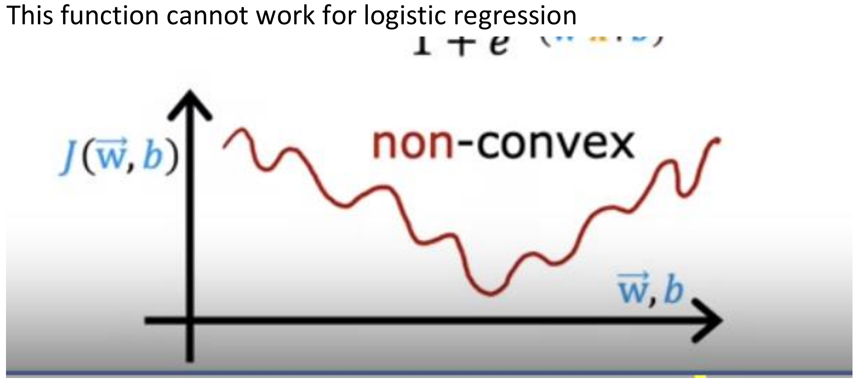
$\boxed{\text{target } y \text{ is } 0 \text{ or } 1}$

$$f_{\vec{w},b}(\vec{x}) = \frac{1}{1 + e^{-(\vec{w}\cdot\vec{x}+b)}}$$

- For this training set how do we choose the parameters *(w and b)*?
  - Squared Error Cost Function
    - $J(\vec{w}, b) = \frac{1}{m}\sum_{i=i}^{m} \frac{1}{2}(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)})^2$
      - For linear regression we get a convex/parabola looking cost function



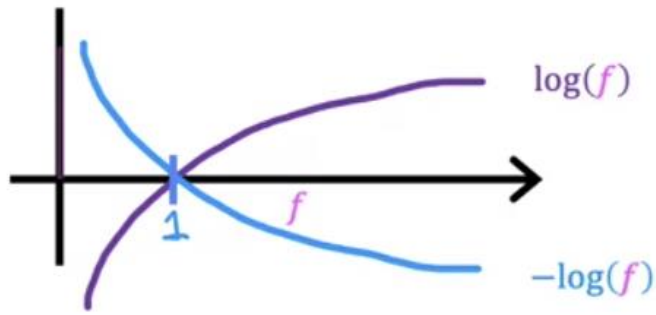
      - This function cannot work for logistic regression



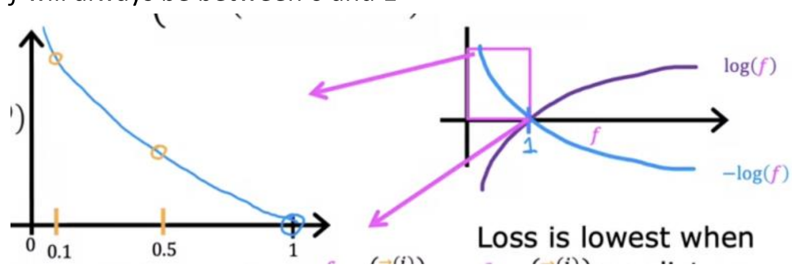
        - Lots of local minima you can get stuck in
  - Logistic Loss Function

$$L(f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}) = \{-log\,(f_{\vec{w},b}(\vec{x}^{(i)}) \text{ if } y^{(i)} = 1\} \text{ or } \{-log(1 - f_{\vec{w},b}(\vec{x}^{(i)}) \text{ if } y^{(i)} = 0)\}$$

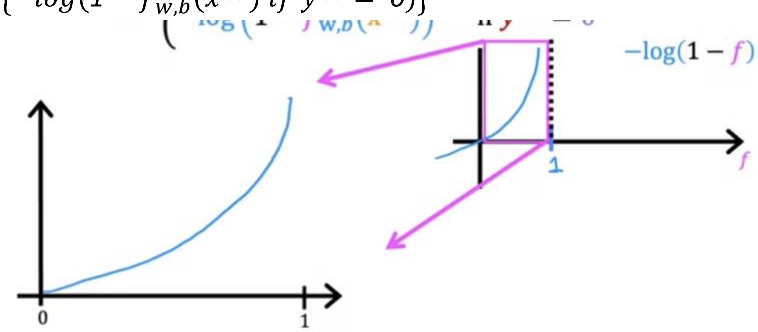
- Plotting -log



  - f will always be between 0 and 1



    - First part of loss function $\{-log\ (f_{\vec{w},b}(\vec{x}^{(i)})\ if\ y^{(i)} = 1\}$
      - If $f_{\vec{w},b}(\vec{x}^{(i)}) \rightarrow 1$ then the loss will be 0 since the algorithm is pretty close
      - If $f_{\vec{w},b}(\vec{x}^{(i)}) \rightarrow 0$ then the loss is ∞ therefore the algorithm pushes to make more better movements
    - Second part of the loss function $\{-log(1 - f_{\vec{w},b}(\vec{x}^{(i)})\ if\ y^{(i)} =\ 0)\}$



      - If $f_{\vec{w},b}(\vec{x}^{(i)}) \rightarrow 0$ then you're very close to optimzing and being close to the true value
      - If $f_{\vec{w},b}(\vec{x}^{(i)}) \rightarrow 1$ then the loss will be ∞ since you're further away from the true label 0

- Simplified Cost Function for Logistic Regression
  - $L(f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}) = \{-log\ (f_{\vec{w},b}(\vec{x}^{(i)})\ if\ y^{(i)} = 1\}\ or\ \{-log(1 - f_{\vec{w},b}(\vec{x}^{(i)})\ if\ y^{(i)} = 0)\}$
    - → Could be simplified to $L(f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}) = -y^{(i)}log(f_{\vec{w},b}(\vec{x}^{(i)})) - (1 - y^{(i)})log(1 - f_{\vec{w},b}(\vec{x}^{(i)}))$
      - If $y^{(i)} = 1$ then →$-log\ (f_{\vec{w},b}(\vec{x}^{(i)})$

$$\boxed{L(f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}) = -\underset{1}{y^{(i)}}\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right) \cancel{- \underset{(1-1)}{(1 - y^{(i)})}\log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right)}}$$

if $y^{(i)} = 1$:

      - If $y^{(i)} = 0$ then → $-log(1 - f_{\vec{w},b}(\vec{x}^{(i)})$

$$\boxed{L(f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}) = \cancel{-\underset{0}{y^{(i)}}\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right)} - \underset{(1-0)}{(1 - y^{(i)})}\log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)})\right)}$$

if $y^{(i)} = 1$:

  - Simplified cost function
    - $L(f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)}) = -y^{(i)}log(f_{\vec{w},b}(\vec{x}^{(i)})) - (1 - y^{(i)})log(1 - f_{\vec{w},b}(\vec{x}^{(i)}))$
      - → Cost function = $J(\vec{w}, b) = \frac{1}{m}\sum_{i=1}^{m} L[f_{\vec{w},b}(\vec{x}^{(i)}), y^{(i)})]$
        - → Expanded Notation:

$-\frac{1}{m}\sum_{i=1}^{m} [y^{(i)}log(f_{\vec{w},b}(\vec{x}^{(i)})) - (1 - y^{(i)})log(1 - f_{\vec{w},b}(\vec{x}^{(i)}))]$

Practice Quiz: Cost Function for Logistic Regression

1. 1 point

$$\overbrace{J(\vec{w},b)}^{?} = \frac{1}{m}\sum_{i=1}^{m} \underbrace{L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right)}_{?}$$

In this lecture series, "cost" and "loss" have distinct meanings. Which one applies to a single training example?

- [x] Loss
- [ ] Cost
- [ ] Both Loss and Cost
- [ ] Neither Loss nor Cost

2. 1 point

Simplified loss function

$$L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right) = \begin{cases} -\log\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right) & \text{if } y^{(i)} = 0 \end{cases}$$

$$\boxed{L\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right), y^{(i)}\right) = -y^{(i)}\log\left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right) - \left(1 - y^{(i)}\right)\log\left(1 - f_{\vec{w},b}\left(\vec{x}^{(i)}\right)\right)}$$

For the simplified loss function, if the label $y^{(i)} = 0$, then what does this expression simplify to?

- ( ) $\log(f_{\vec{w},b}(\mathbf{x}^{(i)})$
- ( ) $\log(1 - f_{\vec{\mathbf{w}},b}(\mathbf{x}^{(i)})) + log(1 - f_{\vec{\mathbf{w}},b}(\mathbf{x}^{(i)}))$
- (●) $-\log(1 - f_{\vec{\mathbf{w}},b}(\mathbf{x}^{(i)}))$
- ( ) $-\log(1 - f_{\vec{\mathbf{w}},b}(\mathbf{x}^{(i)})) - log(1 - f_{\vec{\mathbf{w}},b}(\mathbf{x}^{(i)}))$

Gradient Descent for Logistic Regression

- Gradient Descent Implementation
    - Training logistic regression
        - Goal is to find $\overrightarrow{w}$ and b
            - Given a new $\vec{x}$, output $f_{\vec{w},b}(\vec{x}) = \frac{1}{1+e^{-(\vec{w}\cdot\vec{x}+b)}}$
                - → model can then assess probability that y=1
    - Gradient Descent
        - Cost Function is $-\frac{1}{m}\sum_{i=1}^{m} \quad [y^{(i)}log(f_{\vec{w},b}(\vec{x}^{(i)})) \; - \; (1-y^{(i)})log(1-f_{\vec{w},b}(\vec{x}^{(i)}))]$
            - Algorithm would be

repeat {

j=1...n

$$w_j = w_j - \alpha \underbrace{\frac{\partial}{\partial w_j} J(\vec{w},b)}_{} \rightarrow \frac{\partial}{\partial w_j} J(\vec{w},b) = \frac{1}{m}\sum_{i=1}^{m} (f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)})x_j^{(i)}$$

$$b = b - \alpha \underbrace{\frac{\partial}{\partial b} J(\vec{w},b)}_{} \rightarrow \frac{\partial}{\partial b} J(\vec{w},b) = \frac{1}{m}\sum_{i=1}^{m} (f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)})$$

} simultaneous updates

repeat {

looks like linear regres

$$w_j = w_j - \alpha \left[\frac{1}{m}\sum_{i=1}^{m} (f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)})x_j^{(i)}\right]$$

$$b = b - \alpha \left[\frac{1}{m}\sum_{i=1}^{m} (f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)})\right]$$

} simultaneous updates

- These equations are the same or very similar to the ones we used in linear regression
    - This is not the case though since the definition of f(x) is not the same

Linear regression $\quad f_{\vec{w},b}(\vec{x}) = \vec{w}\cdot\vec{x} + b$

Logistic regression $\quad f_{\vec{w},b}(\vec{x}) = \frac{1}{1+e^{-(\vec{w}\cdot\vec{x}+b)}}$

Practice Quiz: Gradient Descent for Logistic Regression

1. 1 point

Gradient descent for logistic regression

repeat {

$$w_j = w_j - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} \left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right) - y^{(i)} \right) x_j^{(i)} \right]$$

$$b = b - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} \left(f_{\vec{w},b}\left(\vec{x}^{(i)}\right) - y^{(i)} \right) \right]$$

} simultaneous updates

$$f_{\vec{w},b}\left(\vec{x}\right) = \frac{1}{1 + e^{-(\vec{w} \cdot \vec{x} + b)}}$$

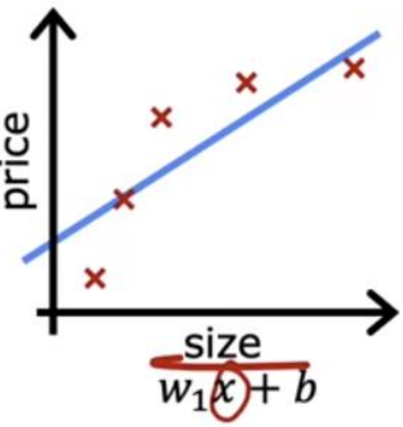
Which of the following two statements is a more accurate statement about gradient descent for logistic regression?

- ○ The update steps are identical to the update steps for linear regression.
- ◉ The update steps look like the update steps for linear regression, but the definition of $f_{\vec{w},b}(\mathbf{x}^{(i)})$ is different.

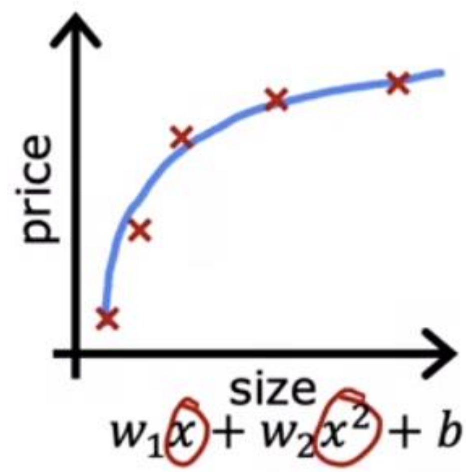
The Problem of Overfitting

- ➢ The Problem of Overfitting
  - What is overfitting and underfitting?
    - Regression Example (House Example)

- While you could do linear regression, it doesn't seem like the best fit

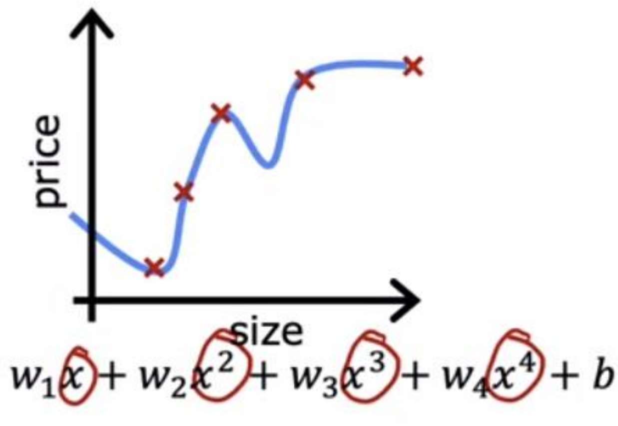


  - The model is **underfitting** the training data or has **high bias**
    - There is a preconception that the housing data will have a linear relationship between the size and the price indicates *high bias*

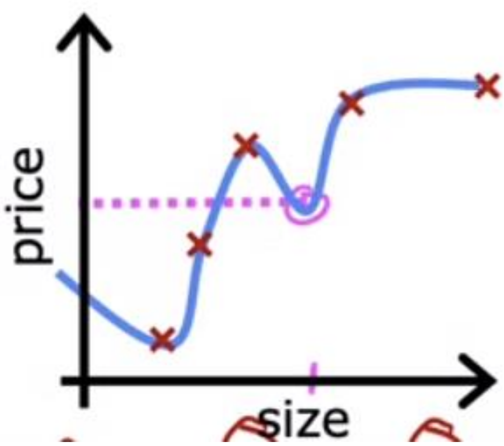


- Fitting a quadratic function
  - Fits the training set pretty well
    - This is good because it allows for **generalization**
      - Generalization means the model would be a good predictor of training samples which are not currently in the training set (brand new examples)

- Polynomial Function

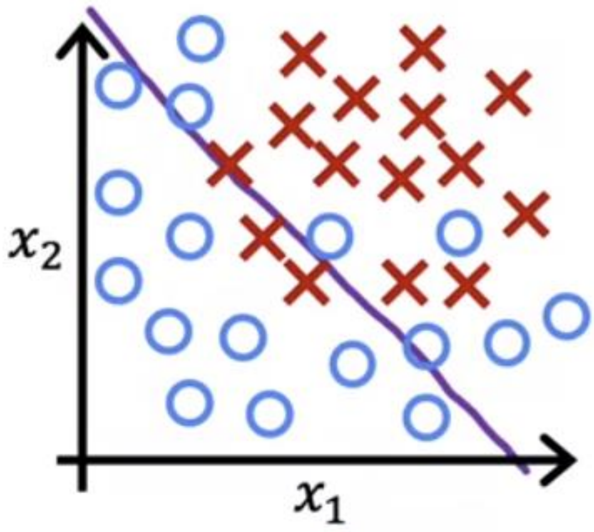


  - Extremely good job at fitting the training data
    - Can get cost function to exactly to 0 since there are no error margins on any training samples
    - The shape of the model used though **overfits** the data as some results where size is bigger will lead to lower prices



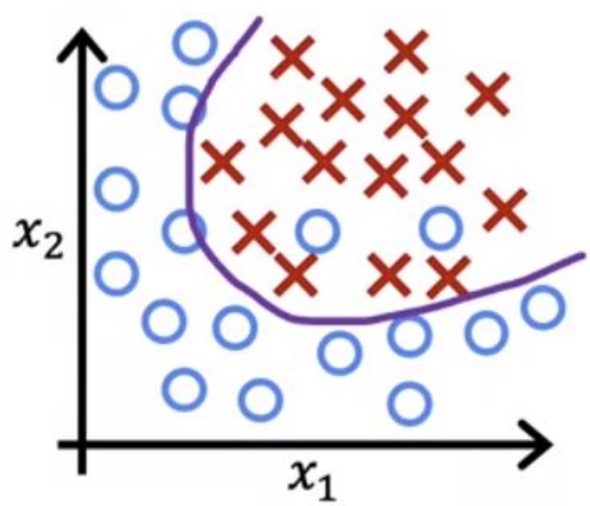
      - Fits the data *too well* and will not likely generalize to new examples
      - Can also be said to have **high variance** (can use overfitting and high variance interchangeably)
- Classification

- Underfitting or high bias

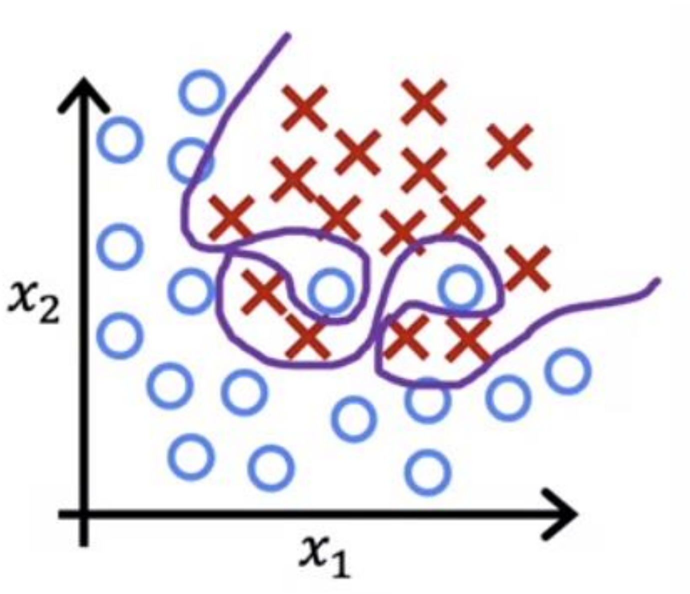


  - While the decision boundary is good, it doesn't look like the best one therefore the function might be underfitting the data

- Pretty good fit to the data

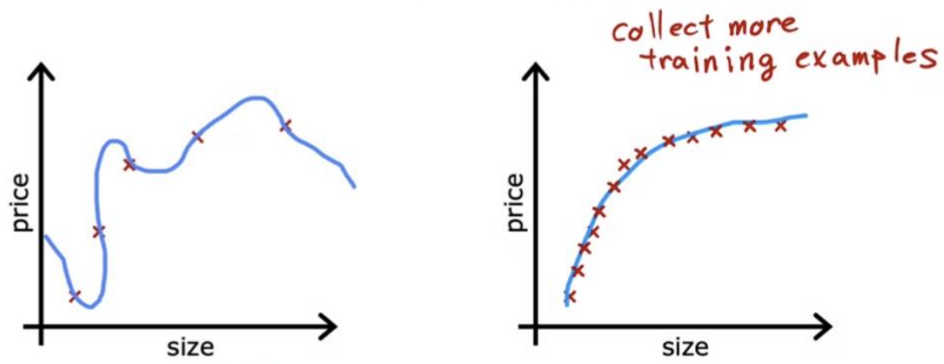


        - Overfit to the data

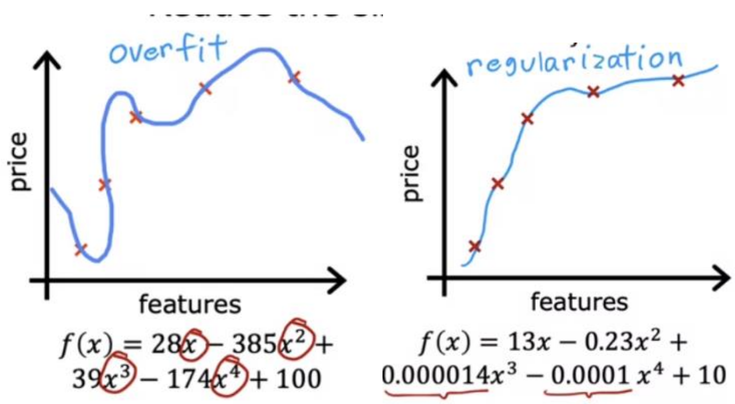


            - Overly complex division boundary that does not allow for generalization

- Addressing Overfitting
    - Collect more training examples - #1 way to address it
        - More data = function that will likely be less wiggly

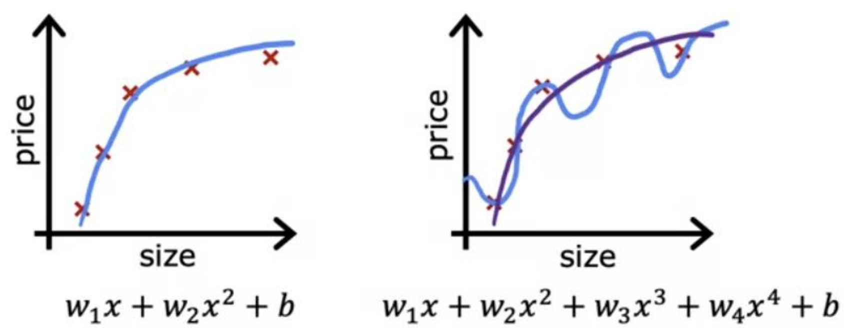


    - Select features to include/exclude
        - If you have a lot of features but not enough training data → Overfitting
        - Picking a subset of the most useful features may be helpful in getting to the "just right" point for fitting a model
            - This is called **feature selection**
                - A disadvantage of this is that the algorithm may be throwing away potential useful features
    - Regularization
        - Way to gently reduce the impact of some features without eliminating it outright
            - Shrinks the value of the parameters without making them 0 and eliminating them entirely



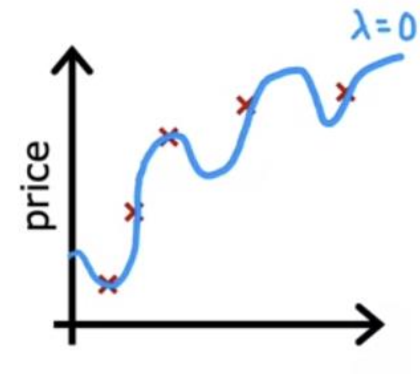
- Reduces the size of parameters w_j to allow for a better fit to the data
  - We usually don't regularize the *b* value

➢ Cost Function for Regularization
- Example

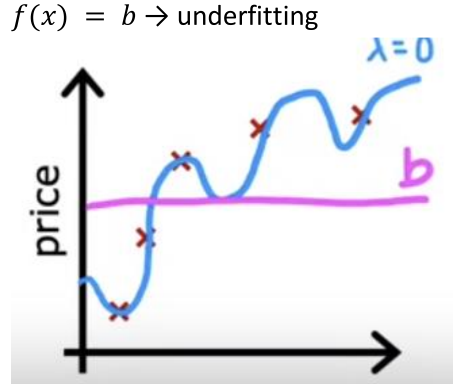


  - Suppose you had a way of making w_3 and w_4 really small (close to 0)
    - If you do this you will have a fit more closely to the quadratic function with the tiny contribution from w_3 and w_4
- If you have n features
  - You might not know which are the important features and which are not
    - In this case we shrink all the w_j parameters
  - Example:
    - You have parameters $w_1, w_2, \ldots w_{100}, b$
      - Penalize all them by adding a term to the cost function
        - →$J(\vec{w}, b) = \frac{1}{2m} \sum_{i=1}^{m} \quad (f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m} \sum_{j=1}^{n} \quad {w_j}^2$
          - Lambda is the regularization parameter
            - $\lambda > 0$
              - If it is 0, you're not using the second term → will continue to have a complex model

(overfitting)

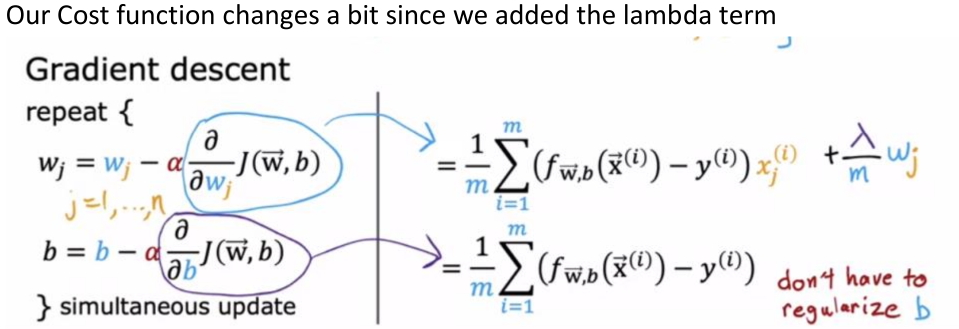


- If it is a big number → placing very heavy weight on the regularization term → will lead all the w_j terms to be approximately 0 → will lead to $f(x) = b$ → underfitting



- Notice both the first and second term are scaled appropriately by 2m
  - Becomes easier to pick a number for lambda
- The first part of the term is referred to as the mean squared error terma and the second part is the regularization term

➢ Regularized Linear Regression
- Our Cost function changes a bit since we added the lambda term



- Implementation

  repeat {

  $$w_j = w_j - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} \left[\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)} \right) x_j^{(i)} \right] + \frac{\lambda}{m} w_j \right]$$

  $$b = b - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)} \right)$$

  } simultaneous update

  - Simplifying the first term

    $$w_j = \underbrace{1 w_j - \alpha \frac{\lambda}{m} w_j}_{w_j \left(1 - \alpha \frac{\lambda}{m}\right)} \underbrace{- \alpha \frac{1}{m} \sum_{i=1}^{m} \left(f_{w,b}(\vec{x}^{(i)}) - y^{(i)} \right) x_j^{(i)}}_{\text{usual update}}$$

    - The second part of the term is the usual update for unregualarized linear regression
    - The first part of the term is usually a number close to 1
      - This is because α and λ are usually small numbers. Divided by a big m value (number of examples in the training set), the value that you are multiplying by w_j is very close to 1
        - For example if you have the value in the parenthesis = 0.998 → every update the value of w_j will shrink just a little bit

➢ Regularized Logistic Regression

- To add regularization to the cost function, you essentially add the same term to the logistic regression cost function as the linear regression cost function

$$J(\vec{w}, b) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log\left(f_{\vec{w},b}(\vec{x}^{(i)}) \right) + (1 - y^{(i)}) \log\left(1 - f_{\vec{w},b}(\vec{x}^{(i)}) \right) \right] + \frac{\lambda}{2m} \sum_{j=1}^{n} w_j^2$$

  - Implementation
    - The only difference is the f(x) function is the sigmoid function applied to z compared to linear regression

      for linear regression:

      repeat {

      $$\underset{j=1\ldots n}{w_j = w_j - \alpha \frac{\partial}{\partial w_j} J(\vec{w}, b)} \quad = \frac{1}{m} \sum_{i=1}^{m} \left[\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)} \right) x_j^{(i)} \right] + \frac{\lambda}{m} w_j$$

      logistic regression

      $$b = b - \alpha \frac{\partial}{\partial b} J(\vec{w}, b) \quad = \frac{1}{m} \sum_{i=1}^{m} \left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)} \right)$$

      }

Practice Quiz: The Problem of Overfitting

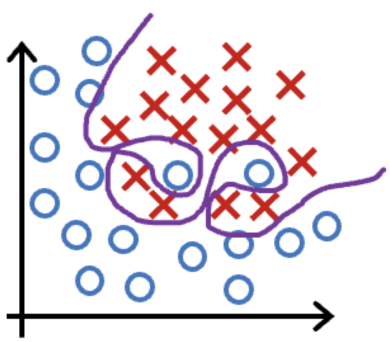
1. Which of the following can address overfitting?

1 point

- [x] Apply regularization
- [x] Select a subset of the more relevant features.
- [x] Collect more training data
- [ ] Remove a random set of training examples

2. You fit logistic regression with polynomial features to a dataset, and your model looks like this.

1 point



What would you conclude? (Pick one)

- ○ The model has high variance (overfit). Thus, adding data is, by itself, unlikely to help much.
- ○ The model has high bias (underfit). Thus, adding data is, by itself, unlikely to help much.
- ○ The model has high bias (underfit). Thus, adding data is likely to help
- ◉ The model has high variance (overfit). Thus, adding data is likely to help

3. # Regularization

1 point

$$\min_{\vec{w},b} J(\vec{w},b) = \min_{\vec{w},b}\left[\overbrace{\frac{1}{2m}\sum_{i=1}^{m}\left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)}\right)^2}^{\text{mean squared error}} + \overbrace{\frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2}^{\text{regularization term}}\right]$$

Suppose you have a regularized linear regression model. If you increase the regularization parameter λ, what do you expect to happen to the parameters $w_1, w_2, \ldots, w_n$?

- [x] This will reduce the size of the parameters $w_1, w_2, \ldots, w_n$
- [ ] This will increase the size of the parameters $w_1, w_2, \ldots, w_n$