ECE 469/ECE 568 Machine Learning

Textbook:
Machine Learning: a Probabilistic Perspective by Kevin Patrick Murphy

Southern Illinois University

August 30, 2024

Minimizing the cost function - Numerical approach

- What happens if our model has two parameters, say w_0 and w_1?

$$f(x, \mathbf{w}) = w_0 + w_1 x, \quad \text{where} \quad \mathbf{w} = [w_0, w_1]$$

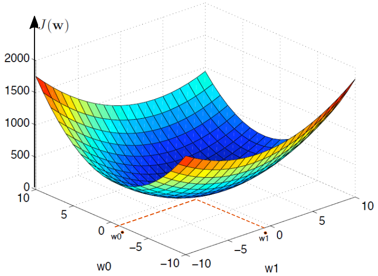
- Then, we need to minimize the cost function over two parameters $\mathbf{w} = [w_0, w_1]$.

$$\min_{w_0, w_1} \left(J(w_0, w_1) = \frac{1}{2} \sum_{i=0}^{m} \left([w_0 + w_1 x_i] - y_i \right)^2 \right)$$

- Again, we can choose different values for w_0 and w_1 and evaluate the cost function.
- Then, we can plot the cost function against w_0 and w_1.
- We are going to see a 3D-plot.

Minimizing the cost function - Numerical approach

$$\min_{w_0, w_1} \left(J(w_0, w_1) = \frac{1}{2} \sum_{i=0}^{m} \left([w_0 + w_1 x_i] - y_i \right)^2 \right)$$



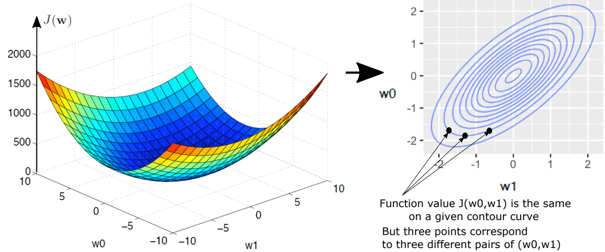
- The cost function is now a surface plot.
- Thereby, the optimal values for w_0 and w_1 which minimize the cost function $J(w_0, w_1)$ can be found from this 3D plot.
- Since this is a convex surface, a global minimum can be found for the cost function. −> Convex functions always have a global minima.

Minimizing the cost function - Numerical approach

- To better understand about a minimization algorithm, we need to dig in deep into the surface plot.
- Any surface (3D) plot has a corresponding contour plot.

Contour plot

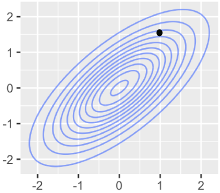
A contour line of a function of two variables (say $J(w_0, w_1)$) is a curve along which the function has a constant value, so that the curve joins points of equal value. It is a plane section of the 3D plot of a function $J(w_0, w_1)$ parallel to the (w_0, w_1)-plane.

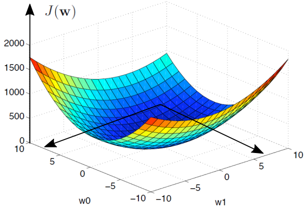


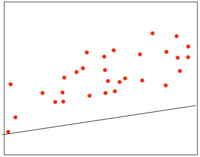
Minimizing the cost function - Numerical approach

$$\min_{w_0, w_1} \left(J(w_0, w_1) = \frac{1}{2} \sum_{i=0}^{m} \left([w_0 + w_1 x_i] - y_i \right)^2 \right)$$

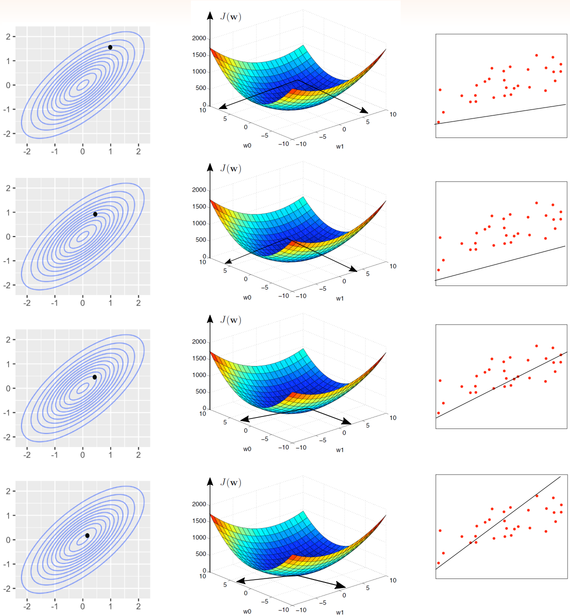
- After an initial choice for (w_0, w_1), we can compute the value of the cost function.
- Then, we need to move towards the center of the contour plot to minimize the cost function by choosing appropriate values for (w_0, w_1).







Minimizing the cost function - Numerical approach



Minimizing the cost function - Gradient descent

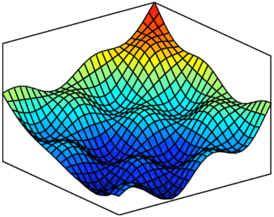
- A summary for minimizing the cost function can be given as

$$\min_{w_0, w_1} \left(J(w_0, w_1) = \frac{1}{2} \sum_{i=0}^{m} \left([w_0 + w_1 x_i] - y_i \right)^2 \right)$$

1. Start with some w_0, w_1
2. keep changing w_0, w_1 to minimize the cost function $J(w_0, w_1)$
3. Continue until we find a minima for $J(w_0, w_1)$

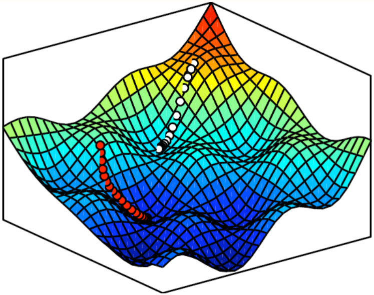
Minimizing the cost function - Numerical approach

- What happens when the cost function is not convex? –> There can be several local minima/maxima together with a global minima/maxima.



- The gradient of $J(\mathbf{w})$ at a point $\mathbf{w}$ can be thought of as a vector indicating which way is "uphill".
- When it comes to an error/cost function, we always like to minimize it by choosing the optimal values for w_0 and w_1.
- Thus, we want to move "downhill" on the surface plot, i.e., in the direction opposite to the gradient or slope.
- This leads to an algorithm which utilizes gradient descent.

Minimizing the cost function - Gradient descent



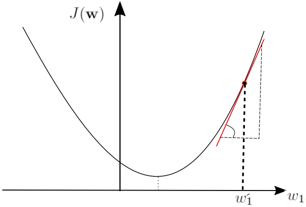
- For more general hypothesis classes, there may be many local optima.
- In this case, the final solution may depend on the initial parameters.
- The algorithm may converge into a local minima depending on the initial choices of $\mathbf{w}$.
- The key idea is to find the steepest gradient descent direction and move towards a local/global minimum.

Minimizing the cost function - Gradient descent

- To obtain intuitions about the gradient descent algorithm, let us again consider a simple cost function with only one parameter.

$$\min_{w_1} \left(J(w_1) = \frac{1}{2} \sum_{i=0}^{m} (w_1 x_i - y_i)^2 \right)$$

- The corresponding cost function can be plotted against w_1 as follows:

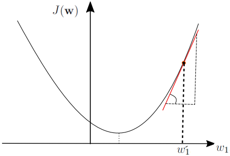


We need to update the value of w_1 as

$$w_1 := w_1 - \alpha \left(\frac{d}{dw_1} J(w_1) \Big|_{w_1 = w_1'} \right)$$

  - Here, α is a positive constant.
  - The gradient/slope at the point w_1' is given by $\frac{d}{dw_1} J(w_1) \Big|_{w_1 = w_1'}$.

Minimizing the cost function - Gradient descent

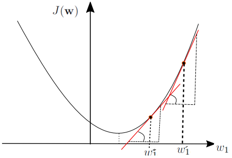


$$w_1 := w_1 - \alpha \left(\underbrace{\frac{d}{dw_1} J(w_1) \Big|_{w_1 = w_1'}}_{>0} \right)$$

- At the point $w_1 = w_1'$, the gradient is positive.
- Hence, the corresponding update will decrease the value of w_1 more towards the minimum of the cost function $J(w_1)$.

Minimizing the cost function - Gradient descent

- Next, we need to evaluate the gradient at the updated value of w_1 (say w_1'').

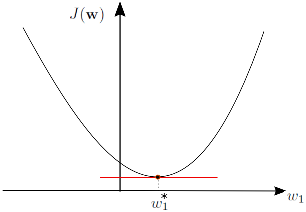


$$w_1 := w_1 - \alpha \left(\underbrace{\frac{d}{dw_1} J(w_1) \Big|_{w_1 = w_1''}}_{>0} \right)$$

- Since the gradient is still positive, this assignment will also move w_1 towards the minimum of the cost function $J(w_1)$.

Minimizing the cost function - Gradient descent

- At the minimum value of $J(w_1)$, the gradient becomes zero.

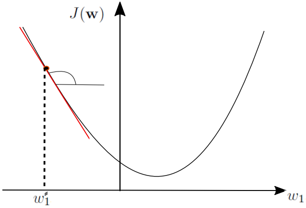


$$w_1 := w_1 - \alpha \left(\underbrace{\frac{d}{dw_1} J(w_1) \Big|_{w_1 = w_1^*}}_{=0} \right)$$

- Hence, we can stop the iteration of the algorithm and report the optimal value of w_1 as w_1^*.
- This optimal w_1^* minimizes the cost function $J(w_1)$.

Minimizing the cost function - Gradient descent

- Now assume that we start at a point that has a negative gradient.



- At $w_1 = w_1^{'''}$, the gradient becomes negative.

$$w_1 := w_1 - \alpha \left(\underbrace{\frac{d}{dw_1} J(w_1)\Big|_{w_1 = w_1^{'''}}}_{<0} \right)$$

- Thus, this update will also increase the value of w_1 towards the minimum value of the cost function $J(w_1)$.

Minimizing the cost function - Gradient descent

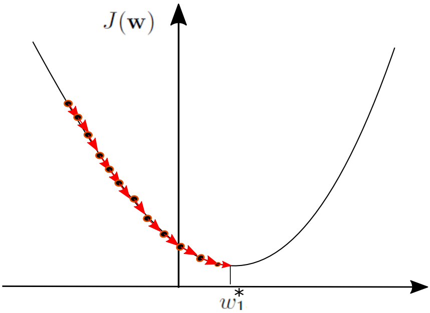
- In gradient descent algorithm, the positive constant α is termed the "learning rate".

$$w_1 := w_1 - \alpha \left(\frac{d}{dw_1} J(w_1) \right)$$

- The learning rate determines how fast your algorithm converges to a minimum.
- However, we need to be careful in choosing a value for the learning rate.

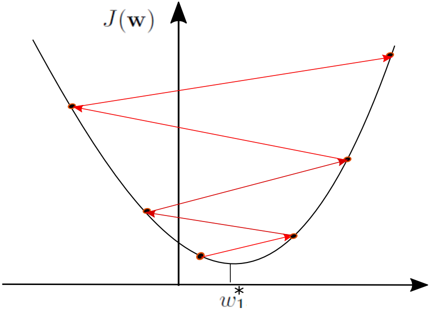
Minimizing the cost function - Gradient descent

- If the learning rate (α) is too small, then our speed of convergence will be much slower –> learning algorithm is slow.



Minimizing the cost function - Gradient descent

- If the learning rate (α) is too large, then algorithm will overshoot –> learning algorithm does no converge to a minimum.



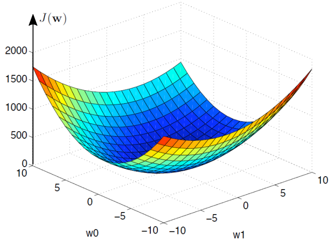
Minimizing the cost function - Gradient descent

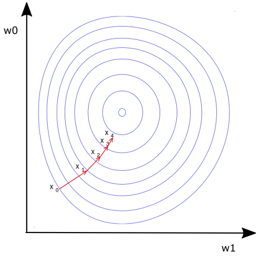
- Let us now investigate how to extend the gradient descent for two parameters in the learning model.

$$f(x, \mathbf{w}) = w_0 + w_1 x, \quad \text{where} \quad \mathbf{w} = [w_0, w_1]$$

- Then the minimization of the cost function becomes

$$\min_{w_0, w_1} \left(J(w_0, w_1) = \frac{1}{2} \sum_{i=0}^{m} \left([w_0 + w_1 x_i] - y_i \right)^2 \right)$$





Minimizing the cost function - Gradient descent

$$\min_{w_0, w_1} \left(J(w_0, w_1) = \frac{1}{2} \sum_{i=0}^{m} \left([w_0 + w_1 x_i] - y_i \right)^2 \right)$$

- A pseudo-code for the gradient descent algorithm can be written as

Gradient Descent Algorithm

Repeat until concergence{

$$w_j := w_j - \alpha \frac{\partial}{\partial w_j} J(w_0, w_1) \quad \text{for} \;\; j = 0 \;\; \text{and} \;\; j = 1$$

}

- The notation := is an assignment/update operator.
- The values of w_0 and w_1 must be simultaneously updated.

$$\begin{aligned} \text{temp0} &:= w_0 - \alpha \frac{\partial}{\partial w_0} J(w_0, w_1) \\ \text{temp1} &:= w_1 - \alpha \frac{\partial}{\partial w_1} J(w_0, w_1) \\ w_0 &:= \text{temp0} \\ w_1 &:= \text{temp1} \end{aligned}$$

Minimizing the cost function - Gradient descent

$$\min_{w_0, w_1} \left(J(w_0, w_1) = \frac{1}{2} \sum_{i=0}^{m} \left([w_0 + w_1 x_i] - y_i \right)^2 \right)$$

- The partial derivatives of the cost function are

$$\begin{aligned} \frac{\partial}{\partial w_0} J(w_0, w_1) &= \sum_{i=1}^{m} \left([w_0 + w_1 x_i] - y_i \right) \\ \frac{\partial}{\partial w_1} J(w_0, w_1) &= \sum_{i=1}^{m} \left([w_0 + w_1 x_i] - y_i \right) x_i \end{aligned}$$

- Thus, the simultaneous update of w_0 and w_1 become

$$\begin{aligned} &\text{temp0} := w_0 - \alpha \sum_{i=1}^{m} \left([w_0 + w_1 x_i] - y_i \right) \\ &\text{temp1} := w_1 - \alpha \sum_{i=1}^{m} \left([w_0 + w_1 x_i] - y_i \right) x_i \\ &w_0 := \text{temp0} \\ &w_1 := \text{temp1} \end{aligned}$$

Linear regression with multiple variables - Gradient descent with multiple variables

- When we have multiple features in our data set, then our model may capture most of these features.

Input variables, Features or Attributes			
House size (x_1)	No. of bedrooms (x_2)	No. of baths (x_3)	Price (y)
500	1	1	100000
1000	2	1	150000
1500	4	2	200000
2000	4	2	250000

- Then our hypothesis needs to be changed to

$$f(\mathbf{x}, \mathbf{w}) = w_0 x_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n = \sum_{l=1}^{n} w_l x_l$$

- Here, $x_0 = 1$ and n is the number of features that the data set possesses.

Gradient descent with multiple variables

- The cost function is given by

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=0}^{m} \left([w_0 + w_1 x_1^{(i)} + w_2 x_2^{(i)} + \cdots + w_n x_n^{(i)}] - y^{(i)} \right)^2$$

- Here, $x_j^{(i)}$ is the jth feature of the ith data entry and $y^{(i)}$ is the output of the ith data entry.
- The minimization of the cost function over parameters/weights is

$$\min_{w_0, w_1, \cdots, w_n} \left(J(\mathbf{w}) = \frac{1}{2} \sum_{i=0}^{m} \left([w_0 + w_1 x_1^{(i)} + w_2 x_2^{(i)} + \cdots + w_n x_n^{(i)}] - y^{(i)} \right)^2 \right)$$

Gradient descent with multiple variables

Gradient descent with multiple variables

$$
\begin{aligned}
&\text{Repeat until concergence}\{\\
&\qquad w_j := w_j - \alpha \frac{\partial}{\partial w_j} J(\mathbf{w}) \quad \text{for} \;\; j = 0, 1, 2, \cdots, n\\
&\}
\end{aligned}
$$

- Then, the simultaneous update of $w_0, w_1, \cdots, w_n$ become

$$
\begin{aligned}
&\text{temp}_j := w_j - \alpha \sum_{i=1}^{m} \left(\sum_{l=1}^{n} w_l x_l^{(i)} - y^{(i)} \right) x_j^{(i)}\\
&\text{continue for all} \;\; j = 0, 1, \cdots, n\\
&w_j := \text{temp}_j \;\; \forall j
\end{aligned}
$$

Gradient descent with multiple variables

- For instance, the updates of w_0, w_1 and w_2 are

$$w_0 := w_0 - \alpha \sum_{i=1}^{m} \left(\sum_{l=1}^{n} w_l x_l^{(i)} - y^{(i)} \right)$$

$$w_1 := w_1 - \alpha \sum_{i=1}^{m} \left(\sum_{l=1}^{n} w_l x_l^{(i)} - y^{(i)} \right) x_1^{(i)}$$

$$w_2 := w_2 - \alpha \sum_{i=1}^{m} \left(\sum_{l=1}^{n} w_l x_l^{(i)} - y^{(i)} \right) x_2^{(i)}$$