Linear Models and Gradient Descent

Class Notes for Machine Learning

1 The Big Picture

A large part of machine learning can be understood as finding a mathematical function that takes some inputs and produces an output.

For example, suppose we want to predict the price of a house.

We might use:

$$x = \text{size of house}$$

and predict:

$$y = \text{house price.}$$

A very simple model might be:

$$\boxed{y = wx + b}$$

where:

- x is the input,
- y is the prediction,
- w is a weight,
- b is a bias.

Machine learning is largely about finding good values for w and b.

We will study several models based on this basic idea.

$$\boxed{\text{Linear Regression} \rightarrow \text{Logistic Regression} \rightarrow \text{Perceptron}}$$

We will also see an important limitation:

$$\boxed{\text{A single linear model cannot solve XOR.}}$$

2 Linear Regression

Linear regression is used when we want to predict a numerical value.

Examples include:

- house price,
- temperature,
- sales,
- salary,
- demand for a product.

The simplest linear regression model has one input:

$$\boxed{\hat{y} = wx + b}$$

The symbol $\hat{y}$ means "predicted y."

For example, suppose:

$$w = 2, \qquad b = 1.$$

If:

$$x = 3,$$

then:

$$\hat{y} = 2(3) + 1 = 7.$$

So the model predicts:

$$\boxed{\hat{y} = 7}.$$

3 Multiple Inputs

Usually we have more than one input.

For example, house price might depend on:

$$x_1 = \text{house size}$$

$$x_2 = \text{number of bedrooms.}$$

The model becomes:

$$\boxed{\hat{y} = w_1 x_1 + w_2 x_2 + b}$$

For three inputs:

$$\hat{y} = w_1 x_1 + w_2 x_2 + w_3 x_3 + b.$$

Using vectors, we can write this more compactly:

$$\boxed{\hat{y} = \boldsymbol{w}^T \boldsymbol{x} + b}$$

This is why linear algebra is useful for machine learning.

4 A Tiny Linear Regression Example

Suppose we have:

$$w = 2, \qquad b = 1.$$

Our model is:

$$\hat{y} = 2x + 1.$$

Consider three observations:

x	y
1	3
2	5
3	7

Our model predicts:

$$\hat{y} = 2(1) + 1 = 3$$

$$\hat{y} = 2(2) + 1 = 5$$

$$\hat{y} = 2(3) + 1 = 7.$$

In this simple example, the model fits the data perfectly.

But real data is usually not this clean.

5 Prediction Error

Suppose the actual value is:

$$y = 10$$

but our model predicts:

$$\hat{y} = 7.$$

The prediction error is:

$$e = y - \hat{y}$$

so:

$$e = 10 - 7 = 3.$$

A common way to measure the error is squared error:

$$\boxed{L = (y - \hat{y})^2}$$

For our example:

$$L = (10 - 7)^2 = 9.$$

The machine learning model wants to find weights that make this error small.

6 Mean Squared Error

With many observations, we can calculate the average squared error.

This is called **mean squared error**, or MSE:

$$\boxed{MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$

The goal of training is:

$$\boxed{\text{Find weights that minimize the error.}}$$

For linear regression:

$$\hat{y}_i = wx_i + b.$$

Therefore:

$$MSE = \frac{1}{n}\sum_i (y_i - wx_i - b)^2.$$

The question becomes:

> How do we find the values of w and b that make the error as small as possible?

One important answer is:

gradient descent

7 Gradient Descent

Gradient descent is a general method for minimizing a function.

Imagine standing on a hill.

You want to get to the bottom.

A simple strategy is:

> Look at the slope and take a small step downhill.

The same idea works for machine learning.

The "hill" is the error function.

The parameters are:

$$w,\ b.$$

We want to move them in the direction that decreases the error.

8 Derivative and Direction

Suppose the error depends on one parameter:

$$L(w).$$

The derivative:

$$\frac{dL}{dw}$$

tells us the slope.

If:

$$\frac{dL}{dw} > 0,$$

the function is increasing as w increases.

To reduce the function, we should move w downward.

If:

$$\frac{dL}{dw} < 0,$$

we should move w upward.

Therefore, we move in the direction opposite the derivative.

This gives the gradient descent rule:

$$\boxed{w_{\text{new}} = w_{\text{old}} - \eta \frac{dL}{dw}}$$

where η is the **learning rate**.

9 The Gradient Descent Weight Adjustment Rule

This is one of the most important formulas in machine learning.

For a weight w:

$$\boxed{w \leftarrow w - \eta \frac{\partial L}{\partial w}}$$

For a bias b:

$$\boxed{b \leftarrow b - \eta \frac{\partial L}{\partial b}}$$

The pieces are:

Symbol	Meaning
w	model weight
L	loss/error
η	learning rate
$\frac{\partial L}{\partial w}$	gradient with respect to w
$\leftarrow$	replace the old value with the new value

The basic recipe is:

$$\boxed{\text{new weight} = \text{old weight} - \text{learning rate} \times \text{gradient}}$$

10 Tiny Gradient Descent Example

Suppose:

$$w = 2$$

and the derivative of the loss is:

$$\frac{dL}{dw} = 4.$$

Suppose the learning rate is:

$$\eta = 0.1.$$

The update is:

$$w_{\text{new}} = w - \eta \frac{dL}{dw}$$

so:

$$w_{\text{new}} = 2 - (0.1)(4)$$

$$w_{\text{new}} = 2 - 0.4$$

$$\boxed{w_{\text{new}} = 1.6}$$

The weight moved from 2 to 1.6 because the gradient was positive.

11 What If the Gradient Is Negative?

Suppose instead:

$$w = 2$$

and:

$$\frac{dL}{dw} = -3.$$

Using:

$$\eta = 0.1,$$

we get:

$$w_{\text{new}} = 2 - (0.1)(-3)$$

$$w_{\text{new}} = 2 + 0.3$$

$$\boxed{w_{\text{new}} = 2.3}$$

Notice that a negative gradient causes the weight to increase.

This is exactly what we want when moving downhill.

12 Gradient Descent for Linear Regression

For a single training example:

$$\hat{y} = wx + b$$

and:

$$L = (y - \hat{y})^2.$$

The derivative with respect to w is:

$$\boxed{\frac{\partial L}{\partial w} = -2x(y - \hat{y})}$$

and the derivative with respect to b is:

$$\boxed{\frac{\partial L}{\partial b} = -2(y - \hat{y})}$$

Therefore:

$$w \leftarrow w - \eta[-2x(y - \hat{y})]$$

and:

$$b \leftarrow b - \eta[-2(y - \hat{y})].$$

These equations show exactly how the prediction error changes the weights.

13 Tiny Weight Update Example

Suppose:

$$x = 2$$

$$y = 5$$

and initially:

$$w = 1, \qquad b = 0.$$

Our prediction is:

$$\hat{y} = wx + b$$

$$\hat{y} = 1(2) + 0 = 2.$$

The error is:

$$y - \hat{y} = 5 - 2 = 3.$$

Suppose:

$$\eta = 0.1.$$

The gradient with respect to w is:

$$\frac{\partial L}{\partial w} = -2x(y - \hat{y})$$

$$= -2(2)(3) = -12.$$

Therefore:

$$w_{\text{new}} = 1 - (0.1)(-12)$$

$$\boxed{w_{\text{new}} = 2.2}$$

The model increases the weight because its prediction was too small.

For the bias:

$$\frac{\partial L}{\partial b} = -2(3) = -6$$

so:

$$b_{\text{new}} = 0 - (0.1)(-6)$$

$$\boxed{b_{\text{new}} = 0.6}$$

After one update, the model is:

$$\hat{y} = 2.2x + 0.6.$$

For $x = 2$:

$$\hat{y} = 2.2(2) + 0.6 = 5.0.$$

In this tiny example, one update happens to produce the correct prediction.

14 Learning Rate

The learning rate controls how large each step is.

The update is:

$$w \leftarrow w - \eta \frac{\partial L}{\partial w}.$$

If η is too small:

learning can be very slow.

If η is too large:

the algorithm can jump past the minimum.

A useful mental picture is:

$$\boxed{\text{learning rate} = \text{size of each downhill step}}$$

15 From Regression to Classification

Linear regression predicts numbers.

For example:

$$\hat{y} = 72.5$$

might represent a predicted house price, temperature, or exam score.

But suppose we want to answer a yes/no question:

spam or not spam?

or:

customer buys or does not buy?

We need a classification model.

This leads to logistic regression.

16 Logistic Regression

Despite its name, logistic regression is primarily a **classification** algorithm.

It starts with a linear model:

$$z = wx + b.$$

Then it passes z through a special function called the **sigmoid** function:

$$\boxed{\sigma(z) = \frac{1}{1 + e^{-z}}}$$

The prediction is:

$$\boxed{\hat{y} = \sigma(wx + b)}$$

The sigmoid converts any number into a value between 0 and 1.

Therefore it can be interpreted as a probability.

17 The Sigmoid Function

Some useful values are:

$$\sigma(0) = \frac{1}{2} = 0.5$$

$$\sigma(2) \approx 0.88$$

$$\sigma(-2) \approx 0.12.$$

So:

$$z \text{ very positive} \Rightarrow \sigma(z) \text{ close to } 1$$

and:

$$z \text{ very negative} \Rightarrow \sigma(z) \text{ close to } 0.$$

For example, if:

$$z = 2,$$

then:

$$\hat{y} = \frac{1}{1 + e^{-2}} \approx 0.88.$$

We could interpret this as:

$$\boxed{P(y = 1 \mid x) \approx 0.88}$$

18 Tiny Logistic Regression Example

Suppose:

$$w = 1, \qquad b = -2.$$

Then:

$$z = x - 2.$$

Suppose:

$$x = 3.$$

Then:

$$z = 3 - 2 = 1.$$

The sigmoid gives:

$$\hat{y} = \frac{1}{1 + e^{-1}} \approx 0.73.$$

So the model predicts approximately:

$$P(y = 1 \mid x) = 0.73.$$

If we use 0.5 as the classification threshold:

$$0.73 > 0.5$$

so we classify the example as:

$$\boxed{y = 1}.$$

19 Why Use the Sigmoid?

A linear model can produce any number:

$$-\infty < wx + b < \infty.$$

That is not convenient if we want a probability.

A probability must satisfy:

$$0 \leq P \leq 1.$$

The sigmoid solves this problem:

$$-\infty \longrightarrow 0$$

$$0 \longrightarrow 0.5$$

$$+\infty \longrightarrow 1.$$

20 Decision Boundary

Logistic regression usually classifies using a threshold.

For example:

$$\hat{y} \geq 0.5 \Rightarrow \text{class } 1$$

and:

$$\hat{y} < 0.5 \Rightarrow \text{class } 0.$$

Because:

$$\sigma(0) = 0.5,$$

the boundary occurs when:

$$wx + b = 0.$$

For one input:

$$x = -\frac{b}{w}.$$

For multiple inputs:

$$w_1x_1 + w_2x_2 + \cdots + w_nx_n + b = 0.$$

This is a **linear decision boundary**.

21 The Perceptron

The perceptron is one of the earliest artificial neural network models.

It also calculates a weighted sum:

$$z = wx + b.$$

But instead of using the sigmoid, it uses a hard threshold.

For example:

$$\hat{y} = \begin{cases} 1 & z \geq 0 \\ 0 & z < 0 \end{cases}$$

The perceptron therefore produces a direct binary answer.

22 Tiny Perceptron Example

Suppose:

$$w = 2, \qquad b = -3.$$

Then:

$$z = 2x - 3.$$

For:

$$x = 2,$$

we get:

$$z = 2(2) - 3 = 1.$$

Since:

$$z \geq 0,$$

the perceptron predicts:

$$\boxed{1}.$$

For:

$$x = 1,$$

we get:

$$z = 2(1) - 3 = -1.$$

Therefore:

$$\boxed{0}.$$

23 Perceptron Weight Update

The perceptron can also learn from mistakes.

A common update rule is:

$$\boxed{w_i \leftarrow w_i + \eta(y - \hat{y})x_i}$$

and:

$$\boxed{b \leftarrow b + \eta(y - \hat{y})}$$

where:

- y is the correct answer,
- $\hat{y}$ is the prediction,
- η is the learning rate.

Notice that this looks similar to gradient-based learning.

The model changes its weights when it makes a mistake.

24 Tiny Perceptron Update

Suppose:

$$x = 2$$

and the correct answer is:

$$y = 1.$$

But the perceptron predicts:

$$\hat{y} = 0.$$

Suppose:

$$\eta = 0.5$$

and:

$$w = 1.$$

The update is:

$$w_{\text{new}} = w + \eta(y - \hat{y})x.$$

Therefore:

$$w_{\text{new}} = 1 + 0.5(1 - 0)(2)$$

$$= 1 + 1$$

$$\boxed{w_{\text{new}} = 2}.$$

The model increases the weight because it predicted 0 when the correct answer was 1.

25 Linear Regression, Logistic Regression, and Perceptron

These models look different, but they share an important structure.

$$\boxed{z = w^T x + b}$$

First calculate a weighted sum.

Then apply a function to it.

Model	**Output**	**Purpose**
Linear regression	z	Predict a number
Logistic regression	$\sigma(z)$	Predict a probability
Perceptron	threshold(z)	Binary classification

This basic structure will become very important when we study neural networks.

26 The Limits of Linear Models

Linear models are powerful, but they have an important limitation.

They can only create a **linear decision boundary**.

For two inputs, the boundary looks like:

$$w_1x_1 + w_2x_2 + b = 0.$$

Geometrically, this is a line.

The line divides the plane into two regions.

Therefore, a single linear classifier can solve problems where the classes can be separated by a line.

27 AND and OR

Consider two binary inputs:

$$x_1, x_2 \in \{0, 1\}.$$

The AND function is:

x_1	x_2	AND
0	0	0
0	1	0
1	0	0
1	1	1

A perceptron can represent AND.

For example:

$$z = x_1 + x_2 - 1.5.$$

If $x_1 = x_2 = 1$:

$$z = 1 + 1 - 1.5 = 0.5$$

so the output is 1.

For the other three cases, $z < 0$, so the output is 0.

Therefore AND is linearly separable.

28 OR

The OR function is:

x_1	x_2	OR
0	0	0
0	1	1
1	0	1
1	1	1

A perceptron can also represent OR.

For example:

$$z = x_1 + x_2 - 0.5.$$

For $(0, 0)$:

$$z = -0.5$$

so the output is 0.

For $(1, 0)$:

$$z = 0.5$$

so the output is 1.

The same is true for $(0, 1)$ and $(1, 1)$.

Therefore OR is also linearly separable.

29 The XOR Problem

Now consider XOR.

XOR means "exactly one of the two inputs is 1."

x_1	x_2	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Notice the unusual pattern.

The positive examples are:

$$(0, 1)$$

and:

$$(1, 0).$$

The negative examples are:

$$(0, 0)$$

and:

$$(1, 1).$$

There is no single straight line that separates the 1s from the 0s.

Therefore:

$$\boxed{\text{A single linear classifier cannot represent XOR.}}$$

30 Why XOR Is Important

XOR is not important because XOR itself is especially useful.

It is important because it demonstrates a fundamental limitation of linear models.

A model of the form:

$$w_1 x_1 + w_2 x_2 + b$$

can only create a straight-line boundary.

But XOR requires a nonlinear boundary.

This was an important insight in the history of neural networks.

31 How Neural Networks Solve XOR

A single perceptron cannot solve XOR.

But multiple perceptrons can work together.

For example, a neural network can have:

$$\text{inputs} \rightarrow \text{hidden layer} \rightarrow \text{output}.$$

The hidden layer can create intermediate features.

Instead of trying to draw one line that solves XOR, the network can combine several simpler boundaries.

Conceptually:

$$\boxed{\text{Several linear transformations} + \text{nonlinear activation} = \text{nonlinear model}}$$

This is the basic idea behind neural networks.

32 Why Nonlinearity Matters

Suppose we have two linear layers:

$$h = W_1 x + b_1$$

followed by:

$$y = W_2 h + b_2.$$

Substitute the first equation into the second:

$$y = W_2(W_1 x + b_1) + b_2.$$

Expanding:

$$y = W_2 W_1 x + W_2 b_1 + b_2.$$

This is still just a linear transformation of x.

Therefore, simply stacking linear layers does not solve the problem.

We need a nonlinear activation function between layers.

For example:

$$h = \text{ReLU}(W_1 x + b_1).$$

Then:

$$y = W_2 h + b_2.$$

The ReLU function is:

$$\boxed{\text{ReLU}(z) = \max(0, z)}$$

The nonlinearity allows the network to represent functions that a single linear model cannot.

33 Putting Everything Together

We can now see a progression.

Linear Regression

Predict a numerical value:

$$\boxed{\hat{y} = w^T x + b}$$

Gradient Descent

Learn the parameters by moving downhill:

$$\boxed{w \leftarrow w - \eta \frac{\partial L}{\partial w}}$$

Logistic Regression

Use a linear model followed by a sigmoid:

$$\boxed{\hat{y} = \sigma(w^T x + b)}$$

Perceptron

Use a linear model followed by a threshold:

$$\boxed{\hat{y} = \begin{cases} 1 & w^T x + b \geq 0 \\ 0 & w^T x + b < 0 \end{cases}}$$

Limitation

A single linear model cannot represent every possible relationship.

In particular:

$$\boxed{\text{XOR cannot be solved by a single linear classifier.}}$$

Neural Networks

Neural networks overcome this limitation by combining linear transformations with nonlinear activation functions.

34 Key Ideas to Remember

1. A linear model has the basic form:
$$\hat{y} = w^T x + b.$$
2. Linear regression predicts numerical values.
3. Gradient descent changes the weights in the direction that decreases the loss:
$$w \leftarrow w - \eta \frac{\partial L}{\partial w}.$$
4. The learning rate η controls the size of each update.
5. Logistic regression uses the sigmoid:
$$\sigma(z) = \frac{1}{1 + e^{-z}}.$$
6. Logistic regression can interpret its output as a probability.
7. A perceptron uses a hard threshold to produce a binary prediction.
8. A single linear classifier creates a linear decision boundary.
9. AND and OR can be represented by a single perceptron.
10. XOR cannot be represented by a single perceptron.
11. Neural networks introduce nonlinear activation functions so that they can represent more complicated relationships.

35 Quick Practice

1. Linear Prediction

Given:

$$w = 3, \qquad b = 2, \qquad x = 4,$$

calculate:

$$\hat{y} = wx + b.$$

2. Weight Update

Suppose:

$$w = 5, \qquad \eta = 0.1, \qquad \frac{\partial L}{\partial w} = 2.$$

Calculate the new weight using:

$$w \leftarrow w - \eta \frac{\partial L}{\partial w}.$$

3. Logistic Regression

Suppose:

$$z = 0.$$

What is:

$$\sigma(z)?$$

4. Perceptron

Suppose:

$$w = 2, \qquad b = -1, \qquad x = 1.$$

Calculate:

$$z = wx + b$$

and determine whether the perceptron predicts 0 or 1.

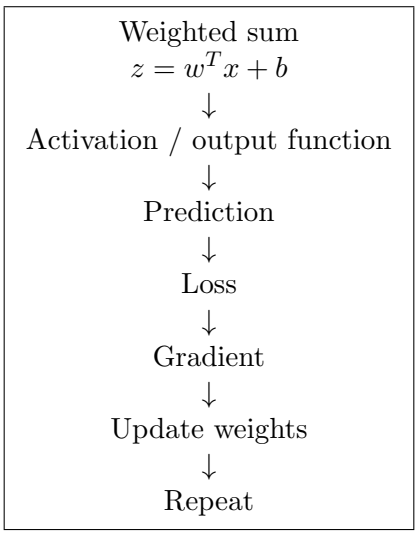
5. XOR

Write the four possible combinations of two binary inputs and their XOR outputs.

Then explain why no single straight line can separate the 0s from the 1s.

36 Final Mental Model

A useful way to remember the material is:



The central idea of machine learning is surprisingly simple:

> Make a prediction, measure how wrong it was, calculate how each parameter contributed to the error, and adjust the parameters to make the next prediction better.

That basic idea leads from linear regression and logistic regression all the way to neural networks.