Lecture 6
Introduction to Machine Learning

Alice Gao

November 2, 2021

Contents

1 Learning Goals **2**

2 Introduction to Learning **2**
- 2.1 Applications . . . 2
- 2.2 Agents that learn . . . 2
- 2.3 The learning architecture . . . 3
- 2.4 Types of learning problems . . . 3
- 2.5 Two types of supervised learning problems . . . 4

3 Supervised learning **5**
- 3.1 High-level ideas . . . 5
- 3.2 Learning as a search problem . . . 5
- 3.3 Example: A prediction task . . . 6
- 3.4 Generalization . . . 9
- 3.5 Bias–variance trade-off . . . 10
- 3.6 Cross-validation . . . 11
- 3.7 Over-fitting . . . 12

1 Learning Goals

By the end of the lecture, you should be able to

- Identify reasons for building an agent that can learn.
- Describe different types of learning.
- Define supervised learning, classification, and regression.
- Define bias, variance, and describe the trade-off between the two.
- Describe how to prevent over-fitting by performing cross-validation.

2 Introduction to Learning

In this lecture, you will be introduced to the area of machine learning.

2.1 Applications

Machine learning has been hugely successful in recent years. Below is just a very small list of applications of machine learning.

- Medical diagnosis: MRI, X-ray, and CT images can be processed by programs which may point out things that may require human attention.
- Facial recognition: smartphones can scan faces to use as a form of ID.
- Handwriting recognition: algorithms which can recognize handwritten digits and letters speed up mail sorting, etc.
- Speech recognition: speech can be processed to talk to various technologies, such as smart home devices and digital assistants.
- Spam filtering: software helps us to filter spam emails based on their content and other properties.

2.2 Agents that learn

Learning is the ability of an agent to improve its performance on future tasks based on experience.

We may want an agent to:

- Do more (expand its range of behaviours)
- Do things better (improve its accuracy on tasks)
- Do things faster (improve its speed)

Why would we want an agent to learn in the first place instead of just programming a solution?

- We cannot anticipate all possible solutions.
- We cannot anticipate all changes over time.
- We may not have any idea how to program a solution (for example, tasks which are easy for humans to do but very hard to know how they are done).

2.3 The learning architecture

Any learning problem has the following key components:

- Problem/task: the behaviour or task we would like to improve our performance on.
- Experiences/data: the data used to improve our performance on the task (usually a sequence of examples).
- Background knowledge/bias: initial knowledge which may bias the initial performance of the agent.
- Measure of improvement: how we can measure performance on the task to track improvement.

2.4 Types of learning problems

We can divide learning problems into three broad categories:

- Supervised learning:

  Given input features, target features, and training examples, predict the value of the target features for new examples given their values on the input features.

- Unsupervised learning:

  Learning classifications when the examples do not have targets defined. Examples include clustering (grouping examples together in a way that makes sense) and dimensionality reduction (projecting high-dimensional data into a low-dimensional space).

- Reinforcement learning:

  Learning what to do based on rewards and punishments. This is somewhere between supervised (exact feedback) and unsupervised learning (no feedback).

In this course, we will focus on supervised learning (with decision trees and artificial neural networks) and reinforcement learning.

Problem: We are given information on user's credit card transactions. We would like to detect whether some of the transactions are fraudulent by finding some transactions that are different from the other transactions. We have no information on whether any particular transaction is fraudulent or not.

Is this a supervised or unsupervised learning problem?

(A) Supervised learning

(B) Unsupervised learning

Solution: The target feature is whether a transaction is fraudulent or not. Since we aren't given the values of the target features, this is unsupervised learning. We are just trying to make some sense of the data on our own.

The correct answer is (B).

2.5 Two types of supervised learning problems

Supervised learning problems can be broadly divided into two categories:

- Classification: target features are discrete.

  For example, classifying the weather as sunny, cloudy, or rainy.

- Regression: target features are continuous.

  For example, determining tomorrow's temperature.

Problem: Is the following problem classification or regression?

You are given historical data on the weather condition (sunny, cloudy, rain, or snow) on a particular day of the year. You want to predict the weather condition on this day next year.

(A) Classification

(B) Regression

(C) This is not supervised learning

Solution: The target feature is the weather condition on a particular day next year. The target feature is one of four different values, so it is discrete. This is a classification problem.

The correct answer is (A).

Problem: Is the following problem classification or regression?

You are given historical data on the price of a house at several points in time. You want to predict the price of this house next month.

(A) Classification

(B) Regression

(C) This is not supervised learning

Solution: The target feature is the price of the house. You might think of the price as discrete since we can really only specify it down to the cent, but usually we treat numerical values like these as continuous values. This is a regression problem.

The correct answer is (B).

3 Supervised learning

3.1 High-level ideas

The basic setting of supervised learning is the following:

- We are given training examples of the form $(x, f(x))$, where x is a vector of input features and $f(x)$ is a vector of target features.
- We return a function h (a. k. a. a hypothesis) that approximates the true function f.

We can assume f exists, but we never get to observe it when solving machine learning problems. All we can hope to do is to approximate it.

3.2 Learning as a search problem

Given a hypothesis space (a space of possible models or representations), learning is a search problem. The goal is to find the hypothesis in this space which best approximates the training examples.

Often, the search space is prohibitively large for systematic search. As a result, machine learning techniques are often some forms of local search.

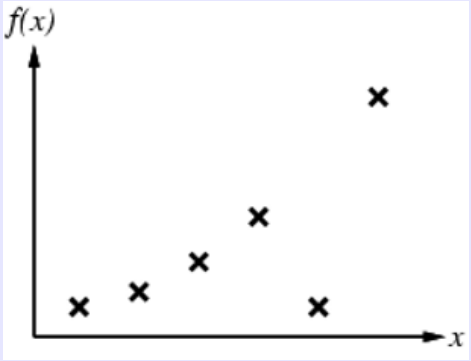
In this interpretation, in order to define a learning problem we need to define the following:

- A search space (of hypotheses)

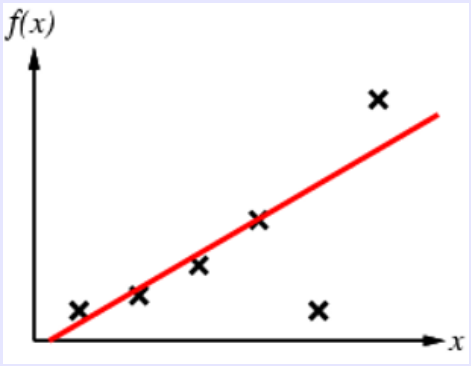
- An evaluation function (used on the hypotheses in the space in order to compare them)
- A search method (how to look through the search space)

3.3 Example: A prediction task

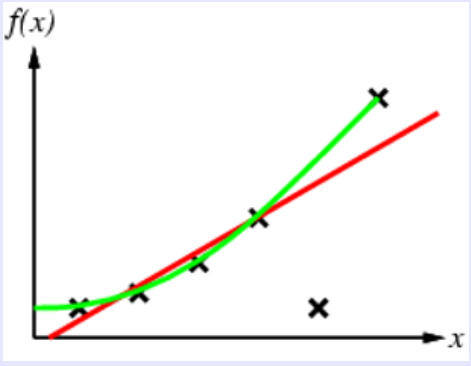
Example: We are given a set of points $(x, f(x))$ and would like to fit a function through these points.



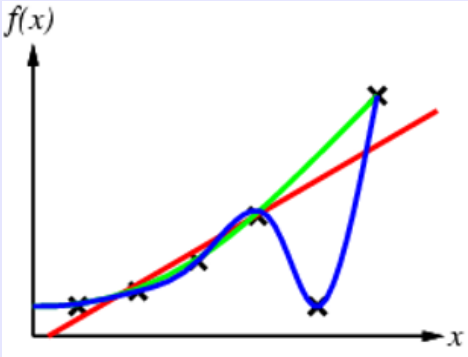
We could use the best linear approximation.



This doesn't seem to go through many of the points. We might not find it satisfying enough. Let's try a slightly more complicated approximation.

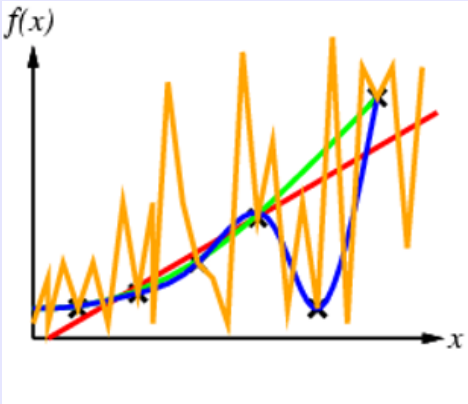


This quadratic function does much better, going through five of the six data points. However, there is still one point left, quite far out from the function. If our goal is to fit a function through the points, we could certainly do better with a higher-degree polynomial, so let's try that.



Now our function is even better—it hits all the data points.

We could take it even further.



This last function is probably too jagged to be a polynomial, but it does go through all the points.

Which one of these functions is the correct one?

In reality, this is a pretty difficult question to answer. The answer really depends on what kinds of assumptions we are willing to make about the data.

For example, if we assume some of the data are outliers (a likely candidate would be the bottom-most point) or the data is generally noisy, then perhaps the first two simple curves are better choices since they don't pay much attention to outliers.

However, if we want a curve that goes through all the points, then the third and fourth curves would be better since they include all the points.

A third interpretation could be that these data points represent stock prices, but our samples are infrequent so the prices don't seem very volatile. If that's the case, then perhaps the last curve is the best choice since it more accurately models changes in stock prices from day to day.

The main message of this example would be that there is no "correct" answer to the question of which function is the best. All curves can be justified as the correct one from some perspective and with some assumptions.

A famous "no free lunch" theorem in machine learning says that to learn something useful, we have to make some assumptions. We have to have an inductive bias.

Some assumptions to consider include:

- Are there outliers in the data? Are some data simply not representative of the underlying function?
- Does the curve follow a particular parametric form? Does it have to look a certain way?

3.4 Generalization

The concrete example in the previous subsection may lead you to believe that machine learning is about finding a function that fits a bunch of data points, but this is not the case. The goal of machine of learning is not to predict all the training examples correctly, but to find a hypothesis that can predict unseen examples correctly.

How is this possible? Usually, the training examples and the unseen examples have some relationship. Perhaps they're all drawn from the same distribution. This relationship can allow us to predict the unseen examples.

If a hypothesis can predict unseen examples well, we say the hypothesis generalizes well. This goal makes machine learning very difficult, but also very exciting.

There are many techniques for choosing a hypothesis that generalizes well. We will discuss two of them.

- Ockham's razor is an assumption that says we should prefer the simplest hypothesis that is consistent with the data. This is also useful in real life—for example, if you are meeting a friend you haven't seen in a while and they are running late, it is probably more likely that they got stuck in traffic than they were involved in an accident.

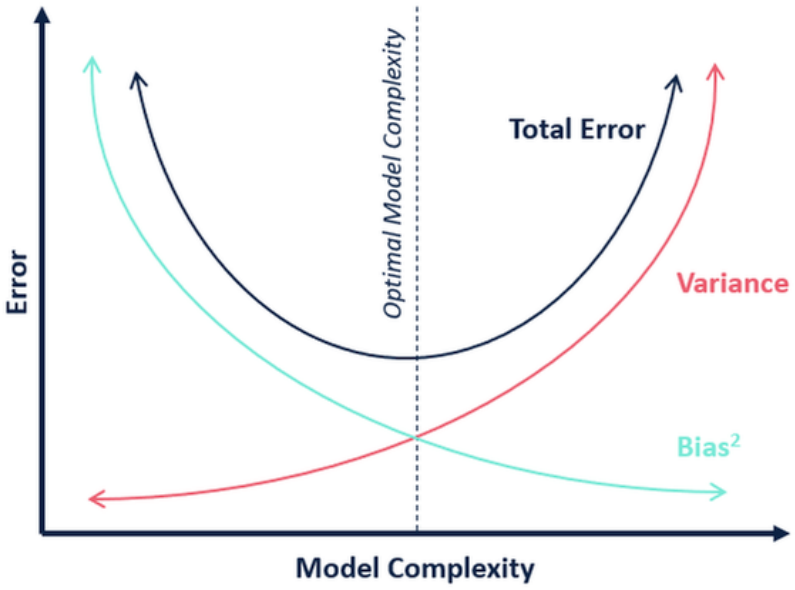
  Ockham's razor is an assumption, so it won't work all of the time, but it tends to work pretty well most of the time.

- Cross-validation is a more principled approach to the problem than Ockham's razor. We will discuss this in more detail later.

Why is it so challenging to find a hypothesis that generalizes well in the first place? The reason is the trade-off between complex hypotheses that fit the training data well and simpler hypotheses that may generalize better. The sweet spot is somewhere in the middle: a hypothesis that isn't too complex but still predicts unseen examples well. This is also called the bias–variance trade-off.

3.5 Bias–variance trade-off

The bias–variance trade-off helps explain how well a hypothesis fits the given data as the hypothesis becomes more complex.



We have the model complexity on the x-axis and the error in the model on the y-axis. This graph shows how the total error changes as the complexity increases: the total error first decreases, then reaches a minimum at some optimal model complexity, then increases again. There are two points where the total error is the highest: one where the model is very simple, and one where the model is very complex.

To understand this behaviour, we can consider how the bias and variance in the model change. As the model becomes more complex, the variance of the model increases and the bias of the model decreases.

- *Bias* in the model describes how well we can fit the data with the learned hypothesis, given infinite data. A hypothesis with high bias usually: makes strong assumptions, is too simplistic, has few degrees of freedom, and does not fit the training data well.

  With high bias, the model limits our ability to fit the data well.

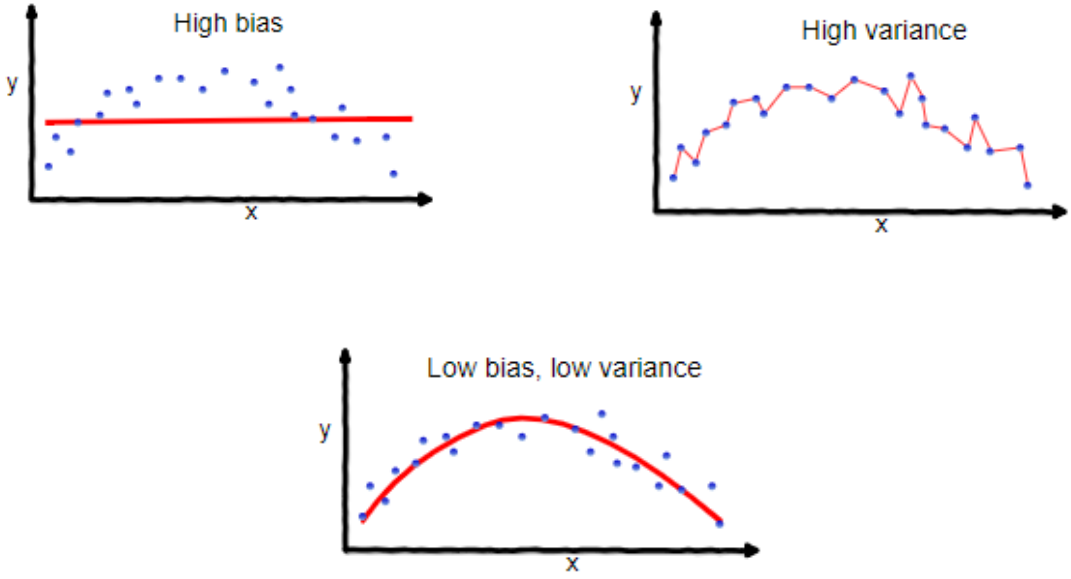
- *Variance* in the model describes how much the learned hypothesis varies given different training data. A hypothesis with high variance usually: has a lot of degrees of freedom, is very flexible, changes a lot whenever the training data changes, and fits the training data very well.

  With high variance, the training data limits our ability to fit unseen data well. This problem is also called over-fitting.

In the graph above, the left-most model has high bias, so we cannot hope to fit the training data well. Meanwhile, the right-most model has high variance, so it may be over-fitting to

the training data at the cost of being able to predict unseen data.

Here is a different set of graphs demonstrating the bias–variance trade-off.



The model with high bias is very simple: a straight line. It makes very strong assumptions and isn't very flexible. If we were to change the data set slightly, the model likely would not change much at all. Clearly, it does not fit the training data very well.

The model with high variance is very flexible. It adapts to all the data points and goes through all of them. However, it will change drastically if we change a few data points.

We'd like to find a model somewhere in the middle with low bias and low variance. We want a model that isn't too rigid and isn't too flexible.

3.6 Cross-validation

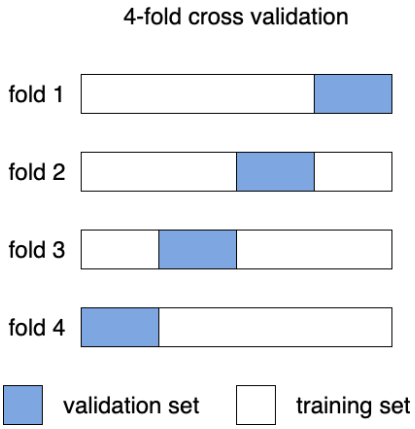
One technique for finding a model with both low bias and low variance is cross-validation. The main idea of cross-validation is that we usually only have training data and no test or validation data, so we will use part of the training set as a validation set.

The steps are:

- Break the training data into K equally sized partitions.
- Train a learning algorithm on $K-1$ partitions (the training set).
- Test the resulting hypothesis on the remaining 1 partition (the validation set).
- Do this K times, each time testing on a different partition.
- Calculate the average error over the K validation sets.

We may have many parameters in the model. In this case, we can perform cross-validation for each parameter.

The following picture illustrates 4-fold cross-validation. Each of the four folds uses a different partition as the validation set, with the remaining three partitions as the training set.



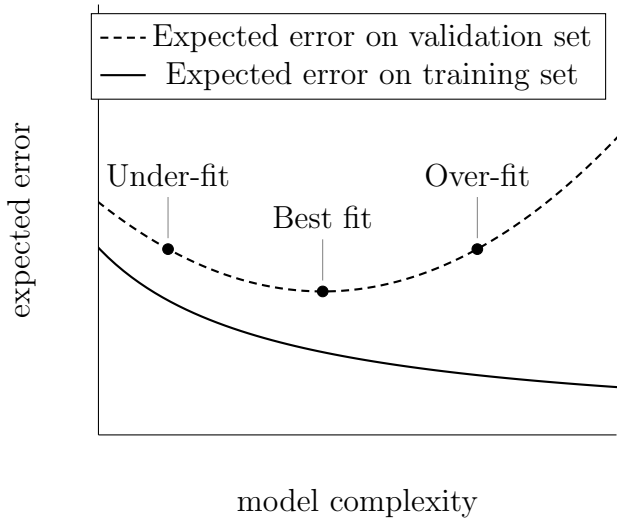
After running cross-validation, we can

- Select one of the K trained hypotheses as the final hypothesis.
- Train a new hypothesis on all of the data, using parameters selected by cross-validation.

3.7 Over-fitting

Over-fitting is closely related to the bias–variance trade-off, but we will discuss it in the context of cross-validation. To perform cross-validation, we split the training data into training and validation sets. How does the expected error change as the model gets more complex?

As the model complexity increases, the error on the training set always decreases. However, this is usually not desirable. Instead, the validation set tells us what to do.



If the model is too simple, we run into under-fitting: the model doesn't have enough degrees of freedom to capture all the useful information in both the training and validation sets. The error on the validation set is large.

On the other hand, if the model is too complex, we run into over-fitting: the model pays too much attention to the training set, capturing characteristics only present in the training set in addition to characteristics shared by the training and validation sets. The error on the validation set is large again.

The best fit is somewhere in the middle: the model has enough complexity to capture the common characteristics of the training and validation sets, but not too much complexity that it over-fits to the training set.