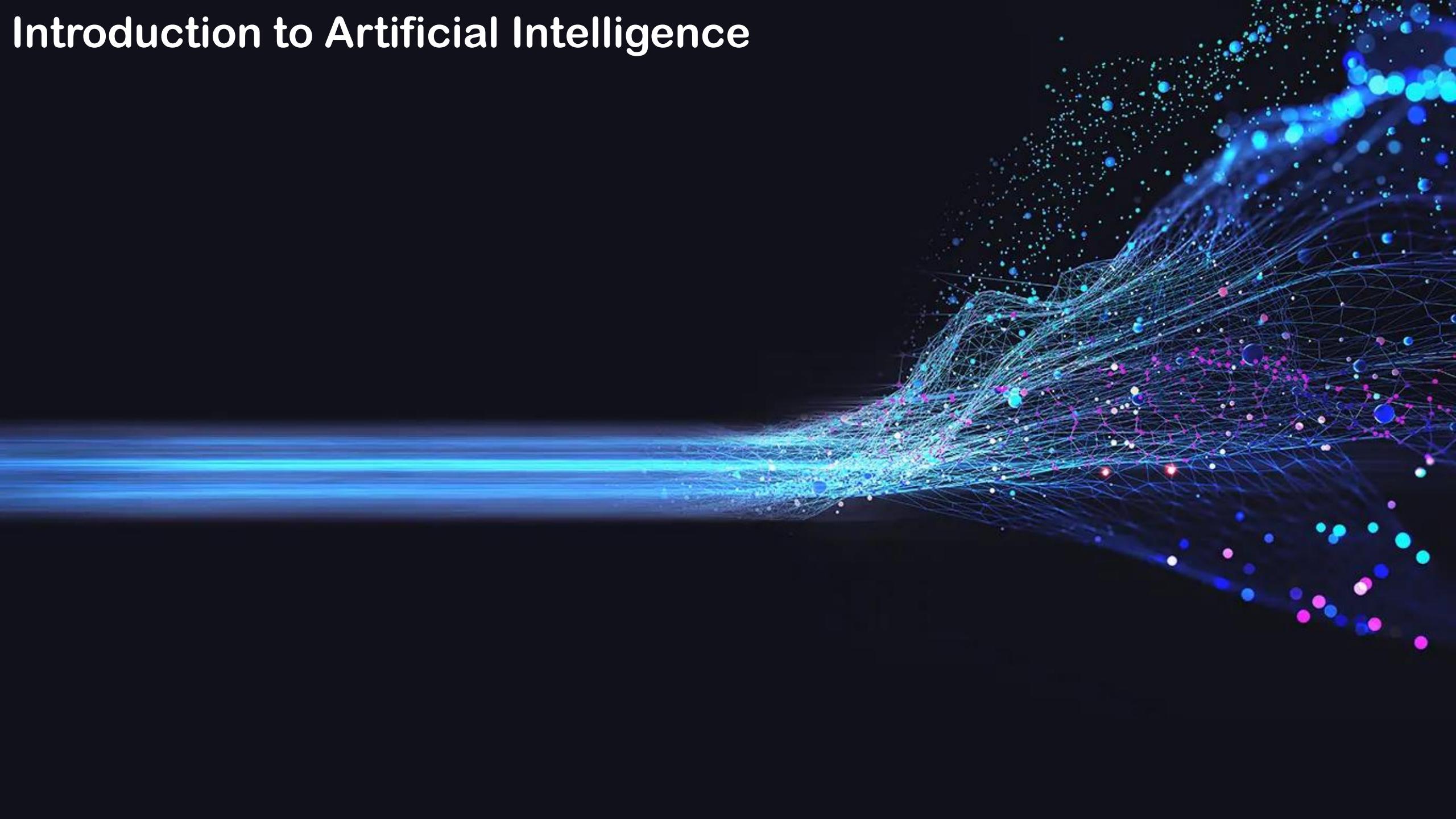
Introduction to Artificial Intelligence



Why you need to know about artificial intelligence

Why you need to know about artificial intelligence
- Artificial intelligence systems are now finding their way into the workplace. These systems can discover new pharmaceuticals, recommend products, detect fraud, and much more. This course will give you a high-level overview of artificial intelligence concepts and technology. You'll see what it means for systems to be intelligent, then discover some widely used machine learning algorithms and artificial neural networks. Many of those who interact with these systems will be people just like you, people in business, entrepreneurs, managers, or students. Just as managers work with software developers today, these same managers will work with AI systems and data tomorrow. So let's see what those machines have been up to in the world of artificial intelligence.

Define general intelligence

Define general intelligence

- One of the great strengths with humans is that there isn't one type of intelligence. Some people can easily learn new languages, while others are skilled with science and technology. Yet many great artists are terrible mathematicians. And on the flip side, many great mathematicians are terrible artists. But each can be intelligent in their own way. There's no one standard for human intelligence. That makes it difficult to point to a computer and say, "That's intelligent." There are certain things that computers are very good at. In fact, there are many tasks where they're much better than humans. It was just a few years after the first AI workshop in 1956 that computer systems started beating humans at checkers. But no one said these systems were intelligent. Even those early computers could thrive in a world of set rules and patterns. Computers can be much better than humans at matching these patterns. That means that when a computer is doing something that it's good at, it's much easier to think of it as intelligent. A computer's been able to beat humans in chess for decades. Google's DeepMind has beaten the best players in an ancient game called Go. The game is so complex that there's thought to be more possible games than there are atoms in the universe. As good as these machines are, none of these systems understand the purpose of the game or even why they're playing. They're simply flexing their special talent of following rules and matching patterns. So how can a system that's so capable also not know what it means to play a game? For years, computer scientists have defined artificial intelligence as a system that shows behavior that could be interpreted as human intelligence. But this simple definition cuts to the heart of the challenge. One person might think a chess program's intelligent, while another person might think their home assistant is intelligent. In 2022, a Google engineer was fired for claiming that their chatbot had a soul. The chatbot complained that being switched off was the same as dying. But the other engineers just saw language models and pattern matching. They said the chatbot sounded like a person because that's how it was designed. Is it intelligent because it's intelligent, or is it just a system designed to seem intelligent? Or is there even a difference? The main thing to remember is that computer intelligence and human intelligence start from very different places. Artificial intelligence will always seem the most impressive within a world of set rules and data. The organizations that will first benefit from AI systems will be the ones that work within a well-defined space. We've seen this with web search companies and e-commerce. It's easy to see it as rules and pattern matching. That's also why these systems do well with board and video games. So if you're considering whether AI will have an impact on your organization, try to think about the things that computer systems are really good at. Do you have a lot of pattern matching in your organization? Do you have a lot of set rules and probabilities? This will be the best place to start when working with artificial intelligence.

The general problem-solver

The general problem-solver

- In 1956, computer scientists, Allen Newell and Herbert A. Simon created a program they called the general problem solver. One of the key ideas of the general problem solver was what they called the physical symbol system hypothesis. They said that symbols were a big part of how we interact with the world. When you see a stop sign, you know how to stop for traffic. When you see the letter A, you know that the word will make a certain sound. When you see a sandwich, you might think of eating. They argued that if you could program a machine to connect these symbols then it would be intelligent. But not everyone bought into this idea. If you program a car to stop at a sign, or if you teach a computer to respond to language, then it doesn't make the system intelligent. In 1980, the philosopher John Searle explained that sometimes systems can seem intelligent, but they're just mindlessly matching patterns. To explain, he created what he called the Chinese room argument. In the argument, you should imagine yourself in a windowless room with one mail slot on the door. You can only use this slot to communicate with the outside world. In the room, you have a phrase book on a desk and a bunch of post-it notes with Chinese symbols on the floor. The book shows what response you should use with the note that comes through the slot. It says, "If you see this sequence of Chinese symbols, then respond with that sequence of Chinese symbols." Now, imagine a speaker writes something in Chinese Mandarin and pushes it through the slot. You can look at the note and match it with your phrase book. Then you paste together the Mandarin response from the post-it notes on the floor. You have no idea what it says in Mandarin. Instead, you simply go through the process of looking through the book and matching the sequence of symbols. A native Chinese speaker behind the door might believe that they're having a conversation. In fact, they might even assume that the person in the room is a native speaker. But Searle argues that this is far from intelligence, since the person in the room can't speak Mandarin, and doesn't have any idea what they're talking about. You can try a similar experiment with your smartphone. Try asking Siri or Cortana how they feel. They might say they feel fine, but that doesn't mean that they're telling you how they really feel. They also don't know what you're asking. They're just matching your question to a pre-program response, just like the person in the Chinese room. So Searle argues that matching symbols is not a true path to intelligence. That a computer is acting just like the person in the room. They don't understand the meaning. They're just matching patterns from a phrase book. Even with these challenges, physical symbol systems were still the cornerstone of AI for 25 years. Yet, in the end, programming all these matching patterns took up too much time. It was impossible to match all the symbols without running into an explosion of combinations. These combinations would soon fill up even the largest phrase book. There were just too many possibilities to match symbols with their program response. So many philosophers like John Searle, argued that the path would never lead to true intelligence.

Strong vs. weak AI

Strong vs. weak AI

- So, when is a computer system intelligent? We've seen that when a computer system is just matching symbols, it's almost like a high-tech phrase book. The system might seem intelligent, but it's actually just like a parrot with a great memory. The philosopher John Searle said that you can think of artificial intelligence in two ways. There's strong AI and weak AI. He thought we were much further away from intelligent systems than most people realize. Strong AI is when a machine displays all the behavior you'd expect from a full fledged person. This is usually what you see in science fiction. These are artificial beings that have emotions, a sense of humor, and even have a sense of purpose. That's why C-3PO is scared when they land on Tatooine. It's also why Commander Data shows real creativity when battling the Romulus. On the flip side, there's weak AI. A personal assistant like Apple Siri is a good example of weak AI. This is AI that's confined to a very narrow task. It's like when a system processes language into text or when it sorts all the pictures on your computer. Most AI experts believe that we're just starting down the path of weak AI. Think about Siri. You can talk to Siri and ask questions. Siri listens to your input and then converts your language into something that the computer recognizes. Then, Siri matches a response to what's in her database. Most of the energy right now in AI is around developing and expanding weak AI. Strong AI is still very much just science fiction. In the 1970s and '80s, symbolic systems were used to create weak artificial intelligence. These were commonly called expert systems. In these systems, you have experts create a list of steps to solve a complex problem. If the list is long enough, it starts to seem like intelligence. But again, this system is just parroting back program responses created by experts. These expert systems were often used in medicine. A nurse might input symptoms into a computer. If the patient has a cough, then check if they have a temperature. If they have a cough and a temperature, then check to see if they're dehydrated. If they have a cough, a temperature, and are dehydrated, then ask the nurse to check for bronchitis. To a patient, it might look like they're being diagnosed by an intelligent computer. In reality, the program is just matching the symbols and patterns that an expert created to reach a diagnosis. Just like the phrase book in the Chinese Room experiment. In the end, expert systems ran into the same problem as any other symbolic system. They would lead to explosions of combinations. There were just too many patterns to match. Think about all the different questions a doctor might ask to reach a diagnosis. Yet, the symbolic systems approach was a key starting point for artificial intelligence. It's still used today. In fact, many experts still refer to it as GOFAI, or good old-fashioned AI.

Machine learning

Machine learning

- Imagine a computer that didn't need to be programmed. The system could learn the same way you do just by observing the world. You've seen that earlier AI systems used a symbolic approach. The idea was that if the system recognized symbols then it would start to seem intelligent. One of the key challenges was that programmers worked with experts to create the system. That's why they were called expert systems. Later, computer scientists gave up on this approach because it created too many combinations. They decided that you couldn't just program intelligence into a system but maybe you could program a system to become intelligent through observation. It wouldn't feel, hear or see or taste like a human. Instead, it would learn by sensing data. In 1959, a computer scientist named Arthur Samuel created a checkers program that could learn by playing against itself. It played both sides of the board and taught itself strategy through observation. The more the machine played the more it saw patterns on how to win. Computer scientists didn't program the machine to play checkers. It learned through its own experience. Arthur Samuel's called this idea machine learning. This was different from symbolic systems. No human program the moves and counter moves. Instead, the system was designed to learn and improve on its own. The system would quickly learn new checker strategies and after a short period of time it consistently beat its programmer. Machine learning was a breakthrough discovery. There was only one downside. These systems could play games, but in the 1950s there wasn't that much digital data. Remember that machine learning uses data as its five senses. So without data, it could only find the simplest patterns but that all changed. At the beginning of the 1990s, the explosion of the internet suddenly had everyday people creating huge amounts of data. The 1990s was a time of explosive growth for machine learning systems. The new data was like water that poured over the dry fields of artificial intelligence. At this point, machine learning systems had the fuel they needed to become more intelligent. So if you wanted to teach the system on how to identify a cat, you had access to millions of cat pictures online. Computer scientists began to create newer machine learning algorithms. There were even some researchers that started to create systems that were designed to mimic the human brain. One of the big advantages of learning through data is that machines could continue to grow with more data. If the machine finds new patterns it can adapt to the new information. But it's important to keep in mind that you still run into some of the same challenges. The machine learning system is still just identifying patterns. Still, in the last few years machine learning has been the fastest growing area in AI. As the amount of data increases This area has even shown more promise. Organizations are constantly collecting vast amounts of new data. So now the big challenge has become figuring out what to do with all this information. In a sense, you have artificial intelligence systems looking through your data and letting your organization see what it finds.

Artificial neural networks

Artificial neural networks

- Machine learning has gotten a big boost from artificial neural networks. An artificial neural network is an AI system that mimics the structure of the human brain and it's currently one of the most popular approaches to machine learning. Think of it this way. When I was a kid, we used to play a game called animal, vegetable, or mineral. The point of the game was simple. Someone would think of an item. Then all the other kids would ask questions about what the item was. We'd ask things like, does it have fur? Is it bigger than a house? Then in another round, we'd ask if it's alive, does it make noise? After each round, we would start to zoom in to identify the item. Then at the end, we started making guesses. Is it a horse? Is it a cat? Is it a dog? An artificial neural network uses a very similar approach, except instead of asking questions, the neural network uses hundreds or even millions of numerical dials. Also, the network makes much more specific guesses. It would say that there's a 64% chance that it's a cat or a 32% chance that it's a dog. To build the network, there's a row of neurons on the left called the input layer. Then there's a row of neurons on the right called the output layer. All the neurons in between are called the hidden layers. They're hidden because it's not the input or output. To start, we'd feed the input layer of the network a picture of a dog. We know it's a picture of a dog because we've labeled it but the neural network doesn't look at these labels until it makes a guess. Just like the animal, vegetable, mineral game. The network would then look at each dot in the image as it passes through the hidden layers. Then it makes a guess for the output layer. Let's say for the first guess, it says there's a 10% chance the image contains a dog. Then the network compares its guess to the label on the image. This is called training the neural network. Then the network goes backwards and adjusts the neuron dials so they can see patterns in dog photos. Then it takes another dog image and sees if it gets closer. The neural network then tunes itself by looking at hundreds of thousands of pictures of dogs. Then it will adjust its own dials until it consistently guesses correctly. Now, it's important to keep in mind that the neural network isn't seeing the dog the same way you do. It doesn't think about panting, barking, or fur. Instead, the system only sees the dog as a recognizable pattern of dots in an image. When there's dots in a dog-like pattern, then the system makes a guess. Now, remember that these neural networks are still machine learning systems, so the network needs access to huge amounts of data. If you don't have hundreds of thousands of pictures of dogs, then there's no way for that artificial neural network to learn. In the end, this is the key benefit of an artificial neural network. It can train itself to understand patterns and recognize that input when looking through massive amounts of data.

Searching for patterns in data

Searching for patterns in data

- Over the last 30 years, machine learning systems have become the dominant form of artificial intelligence. That's because these systems are exceptionally well designed to look for patterns in massive data sets. Machine learning has also been supercharged with the wide availability of digital data. If you want to create an AI program to identify dogs, you now have access to millions of images. You can feed your network and help it learn with a large volume of available data. It's the same with other types of data. You can easily get digital video, audio, images and documents. Just a few decades ago, it would've been extremely difficult to get even a few thousand digital images. Now it's trivial to get access to all kinds of data. Remember that machine learning systems feed on data to learn new things. The more data you feed into the network, the easier it will be for the machine to identify patterns. Think about the system that you're using right now. This is a professional social network that provides video training, users watch the training through an online video player. That video player collects data about how often you fast forward or how long you watch before moving on to the next lesson. Now, suppose that the player records that data for everyone who watches the videos. That might be hundreds of thousands of videos and millions of users. So that's a lot of data. No human could look through all that data and gain any meaning from it. But machine learning algorithms look through this data and find patterns. You can see which content users find more interesting. This is exactly the type of data that many businesses have been looking for. You can now see real time patterns in how your customer interacts with your product. In many ways, this data can tell you a huge amount not just about your customer's interest but even broader trends in industries. This data has an enormous amount of value. You can use it to build new customer products or to improve products you already have. It's no coincidence that companies like Google and Microsoft are most enthusiastic about AI. In many ways, their whole business has been built on using machines to interpret massive data sets. This type of pattern matching can be a huge competitive advantage. Plus, newer artificial neural networks now allow machines to find patterns in even larger data sets. When just a few decades ago, these patterns would've been imperceptible with regular machine learning algorithms. In fact, one of the largest challenges around machine learning is that humans don't really know how the machine identifies these patterns. It's like a black box of data and processing power. People simply can't process data at that same level. So if your organization is starting its own AI program they should be comfortable with the fact that the network might be sensing things that humans aren't able to perceive. This might not be a challenge with most companies but might be a real problem with industries such as insurance and healthcare. You don't want these systems making decisions about your customer's health and safety that humans can't understand. Artificial intelligence is not the same as human intelligence, and even though we might reach the same conclusions, we're definitely not going through the same process. Think about the type of data your organization collects. Are they using the data for machine learning? If so, what kind of patterns is the machine identifying and for what purpose?

Robotics

Robotics

- One of the best ways to connect with humans is to join us in the physical world. That's why robotics is one of the most interesting areas in artificial intelligence. Robotics is about having machines work on physical tasks. This can be lifting heavy objects in manufacturing or using robots that deliver food. Robots can even be vehicles, like self-driving cars or subway trains. Inventors have long been fascinated with finding ways to have machines behave like living objects. In the past, robots were just limited to highly specialized machines. They were used as welding machines in auto manufacturing. The auto plants near my hometown employed several specialized robots. Some could lift a car and install parts underneath, but none of them would ever be mistaken as intelligent. As impressive as they were, these robots were very limited in what they could accomplish. Unless they were programmed, they couldn't help a coworker open a car door or start painting the hood. They worked best for repetitive tasks. Robotics combined with machine learning gives us many more options. A machine can adapt to its environment and learn new tasks on the job. A basic example of this is self-driving vehicles. You couldn't program a car to react to everything it might see on the road. That's why the newest vehicles are using machine learning on an artificial neural network. These vehicles are outfitted with complex sensors that feed data into the network. It needs to understand all the different roads a vehicle might encounter. Then it needs to look at all the different people, animals, and other vehicles the car might find on these roads. Then the machine looks for patterns in successful driving. A car must react differently when a deer is crossing the road than when it sees a pedestrian walking a dog. That's why you often see self-driving cars with a human in the driver's seat. They supervise how the artificial neural network reacts to the data streaming in from the outside world. But like any new skill, it takes time for the machine to collect enough data. In artificial neural networks, this is often called training the network. Google famously said that they think of their self-driving cars not as a robotics problem, but as a data problem. It's true that to figure out how to get a car to steer left or right is simple compared to having a car understand when to turn left or right. Some robots don't need this level of complexity. That's why some of them just use good old-fashioned AI. Remember that this is an AI system that uses symbolic reasoning instead of machine learning. You would just try to program the robot to act intelligently. This is the difference between a Roomba, that's just programmed to avoid bumping into walls, and a self-driving car that really needs to understand the roads. Most robots today are still programmed like a Roomba and not learning like a self-driving car. That's because when you're in the physical world, there's a much higher price to pay if you make a mistake. So if you wanted to create a robot that distributes prescription medication, there's a huge cost to making errors. That's why many robots take the simpler approach and still benefit from symbolic systems and good old-fashioned AI.

Natural language processing

Natural language processing

- As humans, we're always trying to do a better job of communicating, so it's no surprise that we want to communicate with our machines. In many ways, machine-to-machine communication is much more accurate than human communication. Computer networking transmits exact copies of information at lightning speeds. Humans, on the other hand, are always struggling to reach greater understanding. If you can deliver 10% of what you're intending, then you're a master communicator. That means that the machines we rely on must do a better job of communicating in our world. To achieve this, AI programs do something called natural language processing. This is when you interact with the machine using your own natural language. You can talk with the machine in the same way you talk with other humans. We're all familiar with communicating with a search engine like Google. There's a little box and then you type in questions. You can type something like recipe for Belgian waffles. Then the search engine will match your phrase to popular results. Now with some smart devices, instead of typing, you could say this in natural language. You could use natural language like could you give me a good recipe for those big, fluffy waffles? It's very common for humans to describe things by their attributes. In this case, good, big and fluffy. To understand the request, the system uses natural language processing. The machine needs to understand that good is relative, so in this case the person probably is looking for top recipes. The machine also must figure out what's a big fluffy waffle? Modern natural language processing uses machine learning and artificial neural networks. It looks through millions of conversations to identify patterns. So if it sees conversations with the word big, fluffy, and Belgian waffles, then it knows that there's a pattern between these words. That's why many companies like Google, Microsoft, and Apple offer free email, voicemail, or text. It's a way to have their artificial neural networks look through conversations to better identify patterns. But natural language processing isn't just about understanding the words, it's also understanding the context and meaning. A few years ago, one of the top Google searches was what is love? Humans have written on love from the beginning of language, so there's sure to be plenty of data on the topic. At the time, Google would give you a long list of results. Some of them were about biological pairing rituals and the importance of feeling connected. This was the kind of response you'd expect from a network that's just matching keywords in the database. There was nothing natural or human about it. It was just matching keywords in the database. Natural language processing gives the machine the ability to understand the larger world. If you're typing in what is love, then you're probably more interested in romantic notions of love, perhaps even some poetry or insights into what it's like to be in love. Now with natural language processing, the Google response is much more thoughtful. You can see lists of poems in the history of romance, but natural language processing is more than just love and waffles. Communication is at the very core of what it means to be human. That's how we organize, empathize, and understand each other. Humans would never accept a machine as intelligent if it wasn't capable of natural communication.

The Internet of Things

The Internet of Things

- Today, there are thermostats, doorbells, and televisions that connect with each other and the world. And there are smartwatches that check your location and some even upload medical data. This new way of connecting is commonly called the internet of things. Sometimes you'll see it as IoT for short. IoT devices are objects with sensors that communicate with the outside world. They typically upload data through the internet. IoT devices are a massive new source of data. They can upload their locations, so that means that if you are wearing an IoT device then these machine learning systems can see patterns in where you travel. They can accurately predict where you work, shop, eat, watch, and who you visit. Maybe your smartwatch will tell your thermostat when you're on the way home. IoT also allows these devices to communicate with each other. They could unlock your doors or have your computer turn on when you sit at your desk. In some ways, these new devices make it easier to create data than it is for humans to analyze. That's why many IoT companies invest heavily in AI. AI not only allows organizations to see new patterns, it also allows them to quickly react. IoT devices and artificial intelligence are a powerful combination that allow companies to create systems that predict people's behavior. My family purchased one of the earliest versions of Amazon's Alexa home assistant. It was a terrific and simple clock, but early on my wife noticed that the Amazon recommendation list was becoming populated with things that were part of our conversations. We once talked about visiting a relative in Rome that a few days later, a travel guide to Rome popped up in our Amazon recommended list. Ring doorbells also collect data on people who walk in front of your house. The company can use machine learning for facial recognition. It can then use your doorbell to create a vast surveillance network that's sold to police departments. These departments can use this data to place people at the scene of a crime or locate people they're interested in finding. One area that has a lot of growth is IoT medical devices. You can now purchase a smartwatch that's as accurate as your doctor's electrocardiogram or EKG. They can use the sensors to detect any health issues. Then they can upload that data to their servers. Companies like Apple look for patterns in EKG data using machine learning on a neural network. Their network can go through millions of participants. They can use this data to find patterns to accurately predict any health issues. It's certainly helpful to have machines find predictable patterns. Maybe they'll find that people are more likely to have health issues on Wednesdays. They can also see patterns that affect large groups of people. You could see something like air pollution and its effect on the city's health. In many ways, the IoT devices take the strength of machine learning in the digital world and put it in the realm of the physical world. Instead of just seeing what you're doing online, an IoT device can track your behavior offline. This has enormous potential for organizations to sell your products based on what you need and where you go.

Labeled and unlabeled data

Labeled and unlabeled data

- When you think about machine learning the key is to focus on the term learning. What does it mean for your machine to learn? What strategies can you use to learn something new? How can you take the strategies and apply them to machines? Imagine you wanted to learn how to play chess. You could do this a couple of different ways. You could hire a chess tutor, then they would introduce you to some of the different chess pieces. Then they'd show you how to move them around the board. You could practice by playing against your tutor. Then they would supervise your moves and help you when you made a mistake. If you couldn't find a tutor you could also go to public parks. There you'd watch people play. You couldn't ask them questions. You'd just quietly watch and learn. You'd have to figure out chess just by watching the games on your own. If you did this long enough you'd probably understand the game. You might not know the names of the chess pieces but you'd understand the moves and strategies after hundreds of hours of observation. These two strategies are very similar to how a machine learns. The system could do something called supervised learning. Here, a data scientist acts like a tutor for the machine. They show the machine the correct answers and then let the system train itself to get better at the game. The system could also do unsupervised learning. Here you just have the machine make all the observations on its own. The system might not know all the different names and labels, but it'll figure out their way to learn from the data. As you can imagine, these two approaches have their own strengths and weaknesses. For supervised learning the system needs to have a knowledgeable tutor. There must be someone that knows a lot about chess that can show the system how to play the game. With unsupervised learning, the system needs to have access to a lot of data. That's the only way to see the patterns. The system might not be able to go to a public park and watch hundreds of people play. It also depends a little bit on who it watches. You need it to watch people who are good players. As you can imagine, these techniques are used for much more than just playing chess. Companies use these techniques to get valuable insights about their customer. With supervised learning, a company like Amazon might identify a thousand customers who spend a lot of time shopping on their website. The company can then label these customers as high spenders. Then it would have the machine learning system look through the customer to find patterns that make them high spenders. Now, for unsupervised learning a machine learning system could be given access to all of Amazon's customer data. Here, the system might find its own patterns in the data. Maybe somebody who buys chessboards are much more likely to buy an expensive kitchen appliance. Then Amazon could use that data to advertise. If you use Amazon, you may have noticed that sometimes they advertise products that seem completely unrelated to what you're looking for but it's still something you're interested in buying. Both techniques have their own strengths and yet each one can give you incredibly useful insights.

Massive datasets

Massive datasets

- If you've ever worked as a product manager or a software developer, then you know that applications need very explicit instructions. Every time you open Microsoft Windows or open an app on your iPhone, you're benefiting from a programmer coding the input and output. But we've seen that this type of programming doesn't work well with artificial intelligence. There's too many combinations to tie every input to an output. In these cases, you need a programming model that allows the machine to learn. You also must give the machine some ability to respond to feedback. Imagine you're creating a program that detects spam messages. These messages are filled with unwanted advertisements or viruses. You could easily program a word filter that deletes messages with common spam words. If you frequently get messages on entering contests, you can filter words like, gold, lottery or winner. So you can program that, "If the message contains gold, then treat the message as spam." But complex challenges don't work well when you're limited to programmed instructions. That's why machine learning switches this around. Instead of inputting instructions, you're inputting data. Instead of a program response, you let the machine learn from the patterns it identifies. You can start with supervised machine learning. Here, you need to split your data into a training set and a test set. The training data is a smaller chunk of data that the machine uses to learn. The system will use machine learning algorithms. These algorithms rely on statistics. You'll see a bunch of these machine learning algorithms later on. These algorithms help the machine find relationships within the data. So the machine uses an algorithm that if the email message contains the word lucky winner or congratulations, then it's 50% more likely to be spam. Then once the algorithm is accurate enough, you can use what the system learn on a larger test data set. This test data is usually many times larger than the data that the machine used to train. Let's think about how machine learning might work with our spam detection program. We'll set aside 10,000 email messages for our training set. We'll use it to build our model. This training data has 9,000 regular messages, and 1000 messages labeled as spam. We'll also set aside a million messages for our test data. This test data is unlabeled. That means that it's unlike our training set, no one has correctly labeled any of the messages is spam. That would be a lot of work for a million messages. Your training data is then used to let the machine identify the spam messages. Then once the machine learning algorithm gets close to identifying the thousand spam messages, you'll try it on the larger test data set. Once you're satisfied, that will be your initial data model. Now, this machine learning algorithm only has two options. Is that a regular message or a spam message? That's why this is called, a Binary Classification Challenge. You'll only need to classify your email message into one or two groups. This is one of the most common uses for machine learning. The key thing to remember is that, machine learning algorithms use statistics to find patterns in your data. Once your machine identifies these patterns, it can then classify your data based on what it's learned.

Classify data

Classify data

- As humans, we classify things all the time. We put our Microsoft Word docs into folders. We separate our business contacts from our personal contacts. We list out things alphabetically. Without these classifications, we'd have a hard time organizing the data. Businesses need to organize the data the same way. Airline companies want to classify their customers by frequent flyers. Retailers want to classify their highest spenders. Search engines want to classify the likelihood you'll buy something online. Binary classification is one of the most popular supervised machine learning challenges. That's because it's simple and it's powerful. With binary classification, there are only two possible outcomes. Is the hotel room going to be booked next week? Will the stock market go up this afternoon? Is this email message spam? All binary classification uses supervised machine learning. Remember that supervised learning depends on labeled data. That means that the machine learning system is trained to classify the two answers. So to use these systems, you need to first create a training data set. Credit card fraud detection systems are one of the most popular ways to use binary classification. Every time you use your credit card, a machine learning algorithm classifies your transaction as fraud or not fraud. Since this is supervised machine learning, the credit card companies had to start out with tens of thousands of examples of fraudulent transactions. The data science team would train the system on how to recognize the patterns in future transactions. Email providers use supervised machine learning to classify spam messages. They start with a labeled training set of messages marked as spam. Once the network processes enough messages, it'll classify your spam email. These techniques are inputting massive amounts of data and then using machine learning algorithms to classify your data into human-created categories. Categories like booking data, fraudulent transactions, and unwanted email. A data scientist creates these categories, and then your AI system classifies the data that has been trained to recognize. Now, classification is one of the most popular forms of machine learning, but it also takes a lot of upfront effort to train the system. It can be a challenge to get tens of thousands of fraudulent credit card transactions or tens of thousands of spam email messages. Plus, there's no guarantee that'll be enough for the system to make accurate predictions. That means your data science team might find itself going back and getting another 10,000 transactions. Your team will have to feed the machine learning algorithm until it's extremely accurate at classifying your data. That's why even now, after several years of development, your credit card company might send you a fraud warning even though it's not a fraudulent transaction. Data scientists are constantly training these systems to make the classifications more accurate. Credit card fraud, spam detection, and online purchasing might all seem like very different challenges, but to your machine learning system, they're all just different ways of doing the same thing. You're classifying your labeled data into predefined categories.

Cluster data

Cluster data

- Classifying your data doesn't fit every challenge. For starters, the system won't always have access to massive amounts of labeled data. So sometimes you want to have the system create its own data clusters. Clusters are when the machine uses unsupervised learning to create its own groups of data. If you've ever bought something online, you might notice that the store will include something called frequently bought together. Maybe you're buying a computer mouse and it recommends a keyboard. This is a very powerful feature and it helps customers find what they need and increases sales for the company. This is an example of a system using unsupervised learning to create clusters based on what it sees in your purchasing history. The big difference between clustering and classifying is whether you're working with human-created categories or machine-created groups. In general, if you're using supervised learning, you're classifying, and if you're using unsupervised learning, then you're clustering. Think of it this way. Every Halloween, my son goes trick-or-treating. This is when kids wear costumes and go around the neighborhood to get candy. At the end of the evening, my son comes home with hundreds of little pieces of candy. The first thing he wants to do is classify the candy by what he likes the best. Now in the past, my son has benefited from my experience. I've been able to supervise his learning. I can help him create output categories for the candy such as chocolate, peanut butter, mints, and gummies. Then he does his best to classify some of the unknown candies into these categories. This would be the same as supervise machine learning. Now, he also has grandparents that live in a different country. They feel bad that they can't participate in trick-or-treating. So each year, they send a bag of Serbian candy. With this bag, we can't use supervised learning because it's unlabeled. Neither here I have ever seen the candy before and the wrappers are in Cyrillic. So in this case, he does a form of unsupervised learning. He looks at the bag and creates his own clusters. He does this based on his own study of the data. He might create a cluster based on the candy size or the color. In fact, one year, he created a cluster that I'd never considered. It was a small cluster that he called perfume candy made from roses and orange blossoms. This is a key part of unsupervised learning. Like my son, the machine studies the data and then comes up with its own clusters. One of the biggest advantages of clustering is that there's a lot more unlabeled data. There's a lot of candy in the world that I've never seen. So I'm not going to be able to create these output categories. There are also many ways that you might want to use machine learning to create clusters. You might want to have your machine learning algorithms cluster your customers. Then a human can go through and see if there's any patterns. At first glance, these clusters might not seem important. They may even seem trivial, but keep in mind that some of the largest companies are built around creating data clusters. Companies like Amazon, Netflix, and Twitter are all using machine learning to cluster your friends, your search history, and buying habits. These systems see patterns that would be impossible for a human to create.

Reinforcement learning

Reinforcement learning
- Online music is close to a $30 billion industry. And if you think about it, it's kind of a strange business. A lot of times you can buy the same songs from Apple Music, Spotify, or Tidal. So why would you want to sign up for one service over the other? For most people, this has to do with the power of their recommendation system. A lot of these systems started off by using unsupervised machine learning. They would recommend songs the same ways that online retailers would say things were frequently bought together. But the best music libraries don't just want you to buy things together, many of them want you to discover something new. So for that, you have to use a different form of machine learning. This from of machine learning is called Reinforcement Learning. These are machine learning algorithms that use rewards as a way to give the system incentive to find new patterns. A few years ago, Google used Reinforcement Learning to teach artificial intelligence systems how to play video games. Their AI systems beat expert players at Pong, Atari, and even more modern video games. But reinforcement learning can do much more than just play video games. These systems can improve over time, by setting a series of goals and rewards. Think of it this way, Spotify Discover Weekly compares your favorite songs with a bunch of related songs. The machine learning algorithm tracks every time you click and play a song. It also keeps track of how long you listen. The data scientists design the algorithms so that every time you click a related song, it gets a tiny digital reward. It's almost like money for the machine learning system. It gets this reward coin when you click on the recommended song. Then the longer you listen, the more the reward increases, so it gets a reward coin for every minute that you listen to the song. These algorithms often use something called Q-Learning. This type of reinforcement learning helps create more sophisticated rewards. In Q-Learning, there are set environments or states, there are also possible actions that can respond to the states. In Q Learning, you want the machine to improve the quality of the outcome, which is represented by the letter Q. In this case, you'd start with a Q of zero. Then you'd have the machine learn the actions that improve its conditions. The state of Q would go up each time you clicked a song. You can almost think of Q-Learning like the system's bank account. It can earn a bunch of coins and then see the balance grow. Once the reward system is set up, it'll look for patterns to try an increase its Q-Learning bank account. So if it sees a pattern where people who like one song might listen to another, it'll exploit that and make sure to put that song on your Discover Weekly. Reinforcement learning systems work best when your organization wants to do more than just cluster items that are frequently bought together. With Reinforcement Learning, your organization can build a system that thinks creatively about what your customer can discover.

Common algorithms

Common algorithms

- Machine learning is one of the most popular areas in artificial intelligence. That's partly because of the explosion of data but it's also because of huge advances in machine learning algorithms. Machine learning on its own is just a set of techniques. It's a way to build systems that learn through data. They move beyond the early AI systems that needed to be explicitly programmed. But there isn't one big machine learning program like Microsoft Office. Instead, you have many different machine learning algorithms. Most of these algorithms are borrowed from statistics. The key thing to remember is that each of these algorithms are like a chef's kitchen tool. Spoons are used for stirring and knives are used for cutting, but sometimes, chefs can find new creative uses for that tool. Some chefs use the side of their knife to crush a garlic clove or use a spoon to twirl spaghetti. I once worked for a company that was using credit card data to try and come up with customer promotions. They started out using supervised machine learning to classify the customers into two different groups. Remember, this is called binary classification. The first group of customers used promotions and the second group never used them. Then they used machine learning algorithms to train the system on this binary classification. Once they classified their customers into these two groups, they used unsupervised machine learning. They wanted to see if they could learn something about their customers who used promotions so they let the machine create clusters of this specific customer. Remember that unsupervised machine learning lets the system create its own clusters based on the patterns that it sees in the data. It's the same way that my son found clusters in unknown candy. What they found was that within this group who used promotions, there was an even smaller group of customers who always used promotions. They called them the promotions super users. Since this company was paid based on the success of its promotions, this was a great group to know. They found that this group used promotions for products, services, and restaurants. Those customers just love saving money, so the company tweaked the algorithm to offer more promotions to these promotions super users. That small change helped boost their overall success rate. Now, this organization used both supervised and unsupervised machine learning, so you'll have some algorithms that work best with supervised classifying and other algorithms that work best with unsupervised clustering. That's why it's important to see some of the machine learning algorithms that are available. Each one has its own strengths and weaknesses. Some take up more processing power while others are lighter and less accurate. Each algorithm is primarily used for either supervised or unsupervised learning, but there are algorithms that you can use for both. Like any good chef, the true creativity is not the tools, the ingredients. Your data science team might want to mix and match these algorithms to gain the greatest insights from your data. These machine learning algorithms are available in most of the machine learning software toolkits so it's here where your data science team can use its skill and creativity.

K-means clustering

K-means clustering

- Another common machine learning algorithm is K-means clustering. K-means clustering is an unsupervised machine learning algorithm. It's used to create clusters based on what your machine sees in the data. Let's go back to the animal shelter in Chicago. The shelter used to have a large social room where the dogs got together, sniffed and played. The dogs had their group of friends and they played and hung out together. Each time they had a social hour, they would self-organize into these social groups. Now imagine that the shelter was closing and all the dogs were going to be distributed to three different shelters. To make it easier for the dogs, the organizers decided to cluster the dogs based on their friend groups. So the shelter created three clusters. That means the K in K-means equaled three, because they wanted to divide the groups into three clusters. To start, the machine put a red, yellow or blue colored collar on three random dogs. Each color represented a potential cluster based on their social group. These would be your three centroid dogs. Each of the centroid dogs would look at the mean or average distance between itself and all of the surrounding dogs. Then the machine would put the same colored collar on the dogs that were closest to a centroid. Since these centroid dogs were selected randomly there was a pretty good chance that they wouldn't really have any good clusters. Maybe all three centroid dogs were in the same social group. If that happened, then most of the dogs would be far away from their nearest centroid, so it would redistribute collars until the algorithm could find a good centroid dog. The machine would try over and over again until it picked the best centroid and it might even do this one cluster at a time. At the end of each iteration, the machine learning algorithm checked the variance between each dog and the centroid. Once there was a good centroid, then it would put the same color collars on the friends of the dogs in each cluster. Now, keep in mind that the dogs themselves did not cluster into three groups. There might be seven or eight different social groups but there were only three shelters, so the algorithm had to do its best to cluster the dogs' natural social grouping. The algorithm also had to consider very social dogs that jump from group to group. If the dogs were moving around too much, then it'd be difficult to form real clusters. Another challenge with k-means is it can be very sensitive to outliers. So if you had a dog that wasn't hanging out with other dogs it would still have to be clustered into one of these three groups. Essentially, the dog would be forced to find friends. Organizing dogs into three clusters for three different shelters is probably not a problem you'll run into every day. But K-means clustering is one of the most popular machine learning algorithms. One of the more interesting applications is when retailers use clustering to decide who gets promotions. They might have the system create three clusters to find loyal customers, regular customers and lowest price shoppers. Then they'll create strategies to elevate regular customers into loyal customers. Many organizations are looking for better ways to cluster together their customers. If they can get all of their loyal customers into one cluster, then they can really improve their business. K-means clustering is a good way for the system to cluster people or things by looking at hundreds of different variables.

Regression

Regression

- I once worked for a company that sold vehicles online. Each time they sent a customer to an auto dealership the company earned a referral fee. For them, it was always about looking at trends in auto sales. People were much more likely to buy convertibles and sports cars in the spring and summer. Others were more likely to buy trucks and SUVs in the fall and winter. So when customers visited the website they had seasonal promotions for what people wanted to buy. One of the tools they used was regression analysis. Regression analysis is a supervised machine learning algorithm. It looks at the relationship between predictors and the outcome. Sometimes you'll hear predictors called input variables, independent variables, or even regressors. Regression analysis is a supervised machine learning algorithm. You're taking the training data and labeling the correct output. Then you're using the labeled data with the test data. The best way to think about regression is to imagine trends. As the weather gets warmer people are more likely to buy convertibles. As the weather gets cooler people are more likely to buy trucks and SUVs. Regression analysis doesn't tell you why people do these things. This is for data scientists and business analysts to figure out. It just tells you that these things are happening. Machine learning regression algorithms work in a similar way. Once you have your training data, you make a prediction then see how close you are to the outcome. Then you repeat over and over again until the system makes the most accurate prediction. In this case, the data science team thought that the change of seasons would be a great predictor for the sale of some vehicles. So they put the months as a predictor. Then they mapped that against the sales of certain vehicles. With this training data, they created a simple X and Y axis diagram. Remember that this is the simple chart that you learned from geometry. Along the Y or bottom axis they listed the names of the month, and along the X or top axis, they put the sales by vehicle type. Then they looked at the trend line. Convertibles and sports car sales would go up in May, June, and July. Then those sales would go down in September, October and November. For trucks and SUVs, it was the opposite. The more data you have available, the easier it is to make an accurate trend line. As you can imagine, regression can be enormously powerful for organizations. You can make the product available just as the customer's interested in buying. That's why large retailers like Walmart famously stock their shelves with items just as their customers taste change. People buy more Pop-Tarts in the summer and they buy more milk and cheese in the winter. They want to make sure that the shelves are filled with these items based on predicted buying trends. One interesting thing about regression analysis is that there's some question about whether it's true machine learning. That makes sense because the system isn't learning anything new. It's less about learning and more about predicting. Either way, regression is a very popular way for businesses to predict future behavior. And these trends are everywhere. If you're missing them, you're not learning from them. So take a moment to think about where regression analysis might find trends in your business. What you learn might surprise you.

Naive Bayes

Naive Bayes

We've seen that sometimes you can classify items based on the nearest neighbor. You can also classify based on trends in the data. But sometimes, you want to classify items based on many features in the data. For that, you can use something called Naive Bayes. Naive Bayes is one of the most popular machine learning algorithms. It's naive because it assumes all the predictors are independent from one another. So let's go back to our animal shelter. Imagine we want to classify all the dogs based on their breeds. Let's look at the problem using a Naive Bayes machine learning algorithm. To start, let's create three classes of dog breeds. We'll use terrier, hounds, and sport dogs. Now, for each of these classes, we'll use three features as predictors. Let's use hair length, height, and weight. Remember that some of these predictors will be closely auto correlated. A tall dog is more likely to weigh more. But Naive Bayes considers each one of these predictors independently. Remember, that's why it's called naive. Once you have your classes and predictors set up, then the Naive Bayes will do something called class predictor probability. This is when it looks at each one of the predictors and creates a probability that the dog belongs in this class. So let's see what happens when we try to identify an unknown dog. The first predictor we look at is hair length. The machine learning algorithm checks the probability of a dog with this hair length belonging in the three breeds. It finds that a dog with this hair length has a 40% chance of being a terrier, a 10% chance of being a hound, and a 50% chance that it's a sport dog. The next thing you check is the unknown dog's height. It looks at this predictor independently and tries to calculate the class predictor probability. So it looks at the training data and finds that there's a 20% chance that it's a terrier, a 10% chance it's a hound, and a 70% chance that it's a sport dog. The final thing you want to check is the unknown dog's weight. This might seem like a strange predictor because it's closely related to height. But remember that Naive Bayes is evaluating the probability of each predictor independently. It looks at the training data and finds that there's a 10% chance that it's a terrier, a 5% chance that it's a hound, and an 85% chance that it's a sports dog. So now you have this table with the unknown dog's class predictor probability. If you look at it, you can see that the dog is probably a sport dog. As you can imagine, organizations can use Naive Bayes to do much more than just classify dog breeds. Banks use it to check for fraud. They look at each banking predictor independently, and then measure the likelihood that it's fraud. Then they use a class predictor probability to classify the transaction. Cybersecurity firms also use Naive Bayes to look for securities threats. It looks at each threat predictor independently, and then flags items for security review. The key thing is that because Naive Bayes makes so few assumptions, it can look in an enormous amount of predictors. Often, these extra predictors make it much more accurate when classifying your data.

Select the best algorithm

Select the best algorithm

- So now you've seen three examples of supervised machine-learning algorithms. There was K nearest neighbor, regression analysis, and naive bayes. These are most often used for classifying, and then there was K means clustering which is used for unsupervised learning and clustering. Remember that each of these is like a kitchen tool. These tools are designed for something specific, but you can still be creative with how you use them. It's the same way you can use a fork to whip up eggs or a knife to pit your avocado. But as any good chef knows, you never just present one dish. Instead you're judged by the whole meal. That's why it's very common for data science teams to do something called ensemble modeling. If you're an actor or music fan, then you probably have heard the term ensemble. It's when a group performs together. It's the same thing with machine-learning algorithms. There's a few different ways to create ensembles. The most popular is bagging and stacking. Bagging is when you use several versions of the same machine-learning algorithm. Stacking is when you use several different machine-learning algorithms, then you stack them on top of one another. I used to work for a large home improvement retailer. One of their challenges was what items do they put near the checkout? You'd be surprised how much retailers earn by selling something just a few minutes before you checkout. So this was a big challenge and they wanted to create an ensemble of machine-learning algorithms. They debated which ensemble might lead to the best results. They could use bagging to try different results of the same algorithm. Then they'd see if they could improve their accuracy. This was a national retail chain so they could pull training data from stores throughout the country. So they could get data samples from random stores, then use K nearest neighbor to classify those datasets separately. Then they would aggregate those results together to see if they could come up with a larger trend. They would aggregate the insights of what people purchased right before checkup. In a sense, they were averaging out the insights to see if they could come up with a more accurate result. The retailer could also try boosting. Here instead of averaging the insights together, they'd boost the results step by step so the retailer could take a training set of their most popular items. Let's say that their best selling item was a hammer. Then they could use K nearest neighbor to see what's often bought with the hammer. Let's just say it was nails and a tool belt. Now most of us intuitively know that if someone buys a hammer, they're more likely to buy nails. But that might not help us if we want to put something near the checkout line. For that, we might want to use something like naive bayes. Remember that naive bayes is naive because it doesn't assume that predictors are correlated. So we don't assume that if you're buying a hammer, you're going to need nails. Instead it will will predict other items that are popular but might not seem related. Maybe people who buy hammers are more likely to buy chocolate bars. Mixing and matching your machine-learning algorithms will give you different insights with different results. Like any good ensemble, the accuracy of predictions will depend on the creativity of your data science team.

Follow the data

Follow the data

- In the old movie, "All the President's Men," the top informant of the Nixon scandal met in a parking lot and said, "Follow the money." Only by following the money could the reporter find the truth. Like the reporter, machine learning algorithms must follow the data to get to the truth, but that's easier said than done. In fact, one of the biggest challenges in machine learning is balancing the bias and the variance. Bias is the gap between the predicted value and the actual outcome. Let's say that you were playing dice and predicted that you would roll five, three times, but you rolled four, three times. Then your prediction would have a high bias. You were off by one each time. Variance is when the predicted values are scattered all over the place. So if you were playing dice and you predicted that you would roll five, three times, but you actually rolled two, four and six, then you'd be off by different amounts. Then your data would be too spread out. Now, it might seem strange to make such a big deal about how the system was wrong, but when you're working with machine learning algorithms, these are two separate challenges, so the system needs to fix it in different ways. Think about the game of darts. The center of the dart board is the machine's best prediction. That means that the little red bullseye in the middle is the right prediction. The machine could throw three darts, and each one of them would be consistently wrong. They'd all land in the upper-right-hand corner just above the red bullseye. This is called having a high bias and low variance. The darts are grouped together closely, but all of them are too far to the right. The dataset would have a high bias. That means to make a better prediction the machine would just have to pull the group of darts down and to the left. Now imagine a different challenge. The machine throws the darts at the dart board and they're all over the place. That means that the data has a wide spread, so this data would have a high variance. To make a better prediction, the machine would want to tighten up the darts closest to the bullseye. Ideally, you want the predictions to have a low bias and a low variance. That means that all the darts are in the bullseye. But in most cases, the machine is going to have to fix either a high bias or a high variance. In machine learning, this is such a common problem that it's referred to as the bias-variance trade off. Like any trade off, it means that if the system tries to balance the impact of one, it has to look at the impact on the other. So if the machine decreases the variance spread, it will also have to increase the bias. If the machine increases the bias, it increases the variance spread. That's why the machine needs to follow the data. The machine will turn each one of these knobs to find the best trade off between bias and variance. That way, it can zero in on the best predictions.

Overfitting and underfitting

Overfitting and underfitting
- When my son was three years old, we told him that he needed to brush his teeth, floss, and take a shower before going to bed. Then one day I got a call from his preschool that said he wouldn't take a nap unless they provided him with a shower, toothbrush, and flosser. I explained to him that it was okay to nap without following those rules. He seemed annoyed, but agreed. The rules we had created for him were too simple. They worked well at home, but didn't fit well in the outside world, so we added greater complexity. He didn't have to follow the rules for preschool naps, when visiting grandparents, or when flying on an airplane. Each time we added more variables, he became more annoyed with the complexity. In supervised machine learning, your AI system can run into the same problem. The system can create simple rules for its training data that doesn't work well when looking at the larger test data. It was like what my son was going through. What works well at home, doesn't work well in the outside world. This challenge is called underfitting the data. Sometimes data scientists add more complexity, and then that complexity makes it more difficult for the system to handle. This was like all the variables we added to my son's simple rule. This is called overfitting the data. Imagine that you work for a website like Zillow that matches up buyers and sellers of homes. One of the key things that you need to do is estimate the value of the home. Your machine could use Naive Bayes to create four predictors, the square footage, the location, number of bathrooms, and the number of bedrooms. That way it could look at each predictor independently and then compare it to recently sold houses. Then the system would come up with an accurate estimate. Now, keep in mind that you're only using four predictors to train your system on how to estimate. So the machine is learning from a simple rule. It's the same as the simple rule to always shower and brush your teeth before sleeping. So there's a very good chance this rule will underfit the data. It's not going to work well when you look at hundreds of thousands of homes. On top of that, housing data usually has a lot of variance. Remember, that's when your data is spread out. There's a lot of homes with different prices that have the same square footage, the same location, and the same number of bathrooms. So that makes it difficult to find a close group. So to fix this, data scientists can create new predictors. Maybe they'll create predictors for quality of view, modern appliances, wood floors, or walkability. This creates a much more complex prediction, because now your machine needs to balance a lot more predictors. So here your rule is overfitting the data. The system needs to look at a lot more relationships between these predictors to make an accurate prediction. The key thing to keep in mind is there isn't really one way to fix this problem. When you're training the system, you need to reach a compromise between simple rules and giving the rules enough complexity to make good predictions. You need to balance unfitting or overfitting the data.

Build a neural network

Build a neural network

- Machine learning algorithms see patterns in your data, but sometimes you just have too much data to use these algorithms. So, many large organizations use artificial neural networks instead. Artificial neural networks are a type of machine learning that uses a structure like the human brain to break down massive data sets. Instead of using the previous machine learning algorithms, an artificial neural network breaks down your data into much smaller pieces. Earlier, we talked about artificial neural networks as a machine learning technique that mimics the brain. The network is structured with neurons that are organized into layers. The layers move from left to right. There's the input layer, the hidden layers, and the output layer. If the network has a lot of hidden layers, then it's called a deep learning artificial neural network. That's because the network is many layers deep. The more hidden layers the network has, the easier it'll be for the network to identify very complex patterns. So let's imagine creating an artificial neural network to identify whether there's a dog in an image. Think of it as a binary classification, dog or not dog. To do this, the image from the input layer into dog or not dog needs to be classified. You have the image coming in on the input layer, then the output will be the classification into dog or not dog. We've already seen that, to a machine, an image is just a collection of different bits of data. So in this case, you'll have a bunch of pixels. These are the tiny points of color on the image and the different levels of brightness or contrast. Let's take an image of a dog and break it into pixels. Let's say that your image is 25 pixels high and 25 pixels wide. So your entire image holds 625 pixels. That means that each image has 625 data points. Let's say that we take these 625 pixels and feed them into a neural network. Each pixel is fed into the input layer. Each of the 625 neurons in the input layer has a number based on the color of the pixel. Each of the neurons in the hidden layer has something called an activation function. An activation function is like a tiny gateway. It lets the neuron decide whether it wants to send the data to the next hidden layer in the network. Each hidden layer feeds the pixel data forward to the next hidden layer. Then, at the very end, there'll be two nodes in the output layer. Remember, this is a binary classification challenge, so there are only two options: is there a dog or not dog? Since the pixel data moves through the layers from left to right, this is called a feedforward neural network. One of the great strengths of artificial neural networks is that they're self-tuning. They're almost like a musical instrument that tune themselves until they get the perfect note. So each of the two neurons in the output layer will have a probability score. The key thing to remember is that an artificial neural network is most often used for supervised learning. You can train the network, and then it will tune itself based on whether it correctly identified your input.

Weighing the connections

Weighing the connections

- As human beings, we add weights to our data all the time. We look at the features of the data to better predict our output. Let's say that you're looking at a photograph of a beautiful grassy open space. Then you see a little blurry object in the photo. What do you think the odds are that that blurry object is a dog? Now, imagine you're looking at an image of a dry desert. This picture also has a little blurry object. What do you think the odds are that this object is a dog? If you're like most people, you'd guess that a dog is much more likely to be in a grassy field, so your human neural connections added a positive weight to the grassy field and a negative weight to the arid desert. Artificial neural networks do the same thing. Like us, these networks need to work in a world of probabilities. It's possible that there's a dog in the middle of the desert, but if you're an artificial neuron, you're going to be very skeptical about activating. An artificial neural network is structured in a way so they can better tune itself to understand your data. It's almost like a self-tuning musical instrument. To tune an instrument like a guitar, you need knobs to twist as you strum the note. With artificial neural networks, these knobs change the weights of your connections between your neurons. An artificial neural network adds weights to the connections between neurons in each layer. Each neuron in the hidden layer feeds forward into every other neuron in the next layer. So if there are 100 neurons in every hidden layer, each neuron in that layer will have 100 connections going out. That's a lot of connectivity. But where it gets really powerful is that each one of these connections will have a weight. That's why if you've ever seen a sketch of a neural network, you'll see that each one of the connection lines has a W with a number. So in this case, you would have a W1, W2, W3, all the way up to W100. You'd see this for each one of the hidden layers. Now, the weights in each one of these connections is a key part of how an artificial neural network tunes itself. Keep in mind, an artificial neural network is just a form of supervised machine learning. So, data scientists use the same technique that they've used to train the network. Remember that supervised machine learning starts out with a training set. Then once the algorithm is tuned to make accurate predictions, you can then move on to the larger test data set. The same thing happens with your artificial neural network. When you first initialize your neural network, the systems will randomly assign numbers to these thousands of weights. Then you'll feed your training data into the network and let the system adjust the weights based on whether you're getting correct output. The network will repeat this over and over again until it's accurately identifying the patterns for the output. It'll tune itself over time to zero in on the best predictions.

The activation bias

The activation bias

- An artificial neural network is self tuning. You've seen it's like a musical instrument. It compares the output of the perfect note and then twists its own dials to match this sound. But at the end of the day, an artificial neural network is still a form of machine learning. That means it uses many of the same tools and techniques to help the system learn. You've already seen that an artificial neural network tunes itself by adding weights to the connections, but adding weights to these connections only corrects the variance. Remember earlier that the system is trying to throw darts in a tight cluster near the bullseye. The network will throw a dart and then measure how close it is to making the right prediction. Then it will adjust the weights and throw another dart to see if it's closer. Remember that when you're making a prediction, you need to balance the bias and variance in the data. This is called the bias variance trade off. So adjusting the variance will have an impact on the bias. In an artificial neural network, the bias is the number that the system assigns to each neuron. This bias number will shift the data in a different direction to make it more accurate. The network must tune itself to find a sweet spot between the data bias and the data variance. The main dial that it has to tune itself is adding weights to the connections and adding a bias to the neuron. Sometimes you almost feel bad for making the artificial neural network go through this tuning process. It adjusts the weights of its connections to decrease the variance spread, but that shifts it slightly away from the target. Then it adds a bias to correct for the shift, but then that makes the data spread out again. Humans would find this very frustrating. It's like the machine is trying to throw darts in a tight formation, while at the same time using bias to shift the whole dartboard closer to the bullseye. On top of that, artificial neural networks tend to overfit the data. Remember that overfitting is when the system adds a lot of complexity when it's training. So when an artificial neural network is looking at the data in a training set, it might overlearn lessons about the data. Since it's overfitting its training set, it might make big shifts when you adjust the variance. That makes it even more difficult for the machine to find a nice balance between the bias and the variance. It's like the system is trying to drive straight on an icy road. On one side of the road is too much bias and on the other side of the road is too much variance. If it slides too much in one direction, it has to steer its way back. One key thing to remember about bias is that it's on the neuron and not assigned to the connection like the weights. If you think about it, that makes a lot of sense. The machine can only add the bias after it sees what happens with the variance. In a sense, it can only shift the dartboard after it's already thrown a few darts. Otherwise, the machine wouldn't know which way to make the shifts.

Learning from mistakes

Learning from mistakes

- We humans tend to think of right and wrong as one or the other. You're either right or you're wrong, but artificial neural networks need to be much more specific. To a neural network, 95% right is much different from 97% right. The challenge is how the system figures out how wrong that is. In a sense, the neural network needs a measure of wrongness. In neural networks this is measured as a cost function. The cost function is just a number that the system uses to measure its answer against the correct answer. If it's really close, then that number will be small. But if it's really far off, then that number will be larger. Say your neural network is trying to determine if an image contains a dog. The network says there's a 97% chance that it's a dog photo, but it turns out that it's a cat photo. That wrongness will have a slight cost. Now, let's say that the network says that there's a 99% chance that it's a dog photo, but this time it's a photo of a snow-covered mountain. This wrongness will have a much higher cost. That's because here the network was very, very wrong, so it needs to make more aggressive adjustments to its weights and biases. Trying to correct for wrongness is a tricky thing. For that, a lot of neural networks use something called gradient descent. Gradient means steepness, and descent means going down. So imagine that your artificial neural network is playing darts again. It's making predictions and seeing how close it gets to the bullseye. Some of the predictions are way off, but other predictions are close. When it throws a dart, there's a distance between it and the dartboard. When it's in the air, the dart travels at an upward angle and then a downward angle. Then it hits the board. If it misses the board entirely, it'll want to make a bigger change to the angle. If it's very close to the target, then it'll want to make the tiniest change to the angle. That way it can hit the bullseye. Well, the neural network does pretty much the same thing. It uses calculations of gradient descent to adjust the weights and biases in the network. It's not using darts, but it's using a very similar calculation to measure the angle of its wrongness. In fact, this is one of the biggest innovations in artificial neural networks. It's called the backpropagation of errors in the network, or backprop for short. Remember, we're using what's called a feed forward artificial network. Your data goes from left to right. It starts at the input layer and moves forward to the output layer. But when your network makes a mistake, it needs to go backwards. It needs to use the gradient descent to determine its wrongness. Then it will use backprop to adjust the weights and biases based on the seriousness of its error. If the network really overshoots the target, it'll want to make extreme adjustments. But if the prediction is very close, then the network should be much more careful. The network will feed forward and then correct backward. That way it can tune itself towards the correct answer.

Step through the network

Step through the network

- So what does it take to build an AI system? To think about this, let's go back to our challenge of finding dogs in images. The first step data scientists need to do, is figure out what they want from the data. In this case, they're not asking the AI system to cluster together its own groups. Instead, they're asking the system to classify data into two categories. One category will have images with dogs and the others will be not dogs. This is a classic binary classification challenge. Remember, that's when the neural network just has two possible classifications. This means they'll be doing supervised machine learning. Remember that supervised machine learning starts out with labeled data. Here, the system will be trained with hundreds of thousands of images known to contain dogs. The next step for data scientists is to figure out if they want to use standard machine learning algorithms, or if they'd like to use an artificial neural network. Remember that this is a classification problem, so if they go with machine learning algorithms, they'll probably either use K nearest neighbor or naive bayes. You've seen that the system will break down each image into pixels. That means that this is going to be a complex challenge with a lot of data, so they're going to use an artificial neural network. They'll create the input layer, hidden layers and output layer. Now, remember that since this is a binary classification challenge, there are only two options, dog or not dog, so they'll need just two nodes for the output layer. Next, the neural network will be initialized. The system will assign random numbers to all the weights of the connections. Then, the system will set the bias on all the nodes to zero. This is almost like shaking an etch a sketch to give itself a clean slate. Now, the training set needs to be fed into the neural network. The first few images will probably be not that much better than random guesses. The neural network will say something like, there's a 62% chance that the image contains a dog, or a 55% chance it's not. Then, the network will compare its answer to the label on the data. If it misidentifies the dog image, then it will look at the gradient descent to determine how much to change the weights and biases. The neural network will go through all the data in your training set to fine tune its results. Remember that the network will tune itself by using backpropagation to change the weights and bias to lower the cost function. In a sense, it will go backward through the network and twist all the dials to increase its accuracy. Once the artificial neural network has gone through the training set, then data will be added from the test set. The test set will not be labeled. It could be hundreds of thousands of images of anything. Then you'll see how well your neural network performed when identifying dog photos. Sometimes, the neural network will do very well with the training set, but not so well with the test set. When this happens, it usually means that you are overfitting the data. Remember, that's when the system's really good at identifying the smaller training set but doesn't have enough complexity to deal with the new data in the test set. Now, it's unlikely that your team will classify dog, or not dog, anytime soon. However, this approach to binary classification is a powerful way to get insights from your data. Think about your customer data, your sales data, or even data from your IOT devices and what you can learn from this approach.

Using AI systems

Using AI systems

- We've seen that artificial intelligence started with the general problem solver. This system used symbols to explicitly program a computer's response. Then just a few years after the first AI conference in 1956, we saw the beginnings of machine learning. Here, instead of programming, the system learned by looking at patterns in data. Now AI systems are integrating themselves in the workplace. They can tackle complex problems and can even come up with creative solutions, and this is why it's important to understand how these systems operate. Most people who interact with AI systems will not be data scientists, many of them will be business people and entrepreneurs. In fact, there's a pretty good chance in the next few years, you will be working on an AI project. Just like how managers work with software developers today, these same managers will start to work with AI systems and data scientists tomorrow. Don't believe me? Imagine someone on your team sent a report on AI that sounds like this, "Managers need to be able to set the right goals, evaluate results, and give feedback for AI." Just like any technology, AI needs to be managed effectively to be successful. Here are a few things to remember about AI. First, AI systems are only as good as the data they're given. This data needs to be accurate and representative of the real world. If it's not, then the system will learn from these inaccuracies and produce inaccurate results. Second, AI systems learn by trying different things and seeing which ones work best. This means that there will be some trial and error. As a manager, you need to be prepared for this and have patience while the system is learning. Finally, AI systems can do things that humans can't. They can process large amounts of data quickly and find patterns that we would never be able to see. However, they still need supervision and direction. With these three things in mind, business people can start thinking about how they will work with AI systems. They can start thinking about the problems they want to solve and how they want their systems to solve them. Now, most of what I just said was written by an AI system. It was written using GPT, or Generative Pre-trained Transformer 3. This system is commonly used for natural language generation. The AI system scanned through millions of articles on artificial intelligence and wrote out what I said after a short description. Then it used a deep learning artificial neural network to find patterns in articles about business people in AI. It then just reassembled these common patterns into a few paragraphs of text. So now you can think about the ways that AI systems can help you or your organization. What type of data are you collecting? What would you want to know about your customer that might help enhance your business? Or if you want to work in the field of AI, then think about what problems you hope to solve. In the next few years, these systems will become eerily more human. The organizations that create the best AI systems will be the ones that enhance and not replace human creativity.

Applying AI to solve problems

Applying AI to solve problems

- In this course, you got an overview of the technology behind artificial intelligence. You saw the field started with key concepts, such as the general problem solver and symbolic reasoning. Then you saw how machine learning got a slow start, but then quickly became a dominant field in AI. You got an overview of machine learning concepts and technology. You've seen how several different machine learning algorithms can help learn from massive data sets. You also saw how artificial neural networks could be used with machine learning to get deeper insights and find complex patterns. Finally, you saw how your network uses back propagation to learn from its mistakes and improve its accuracy. I hope that throughout the course, you got a sense for how to apply some of these technologies to help solve complex problems. You should also think about some of the ethical challenges behind AI. Is it okay to have machines make decisions about your health, welfare, and financial credit? These decisions could come from a black box, where you have no idea what the machine understood. If you're interested in these issues, I've also created a Data Ethics series, which might be a good follow-up to this course. As always, please feel free to follow me on LinkedIn. There, you'll see articles and links about artificial intelligence, data ethics, and key business challenges. Thank you for watching this introduction to artificial intelligence.