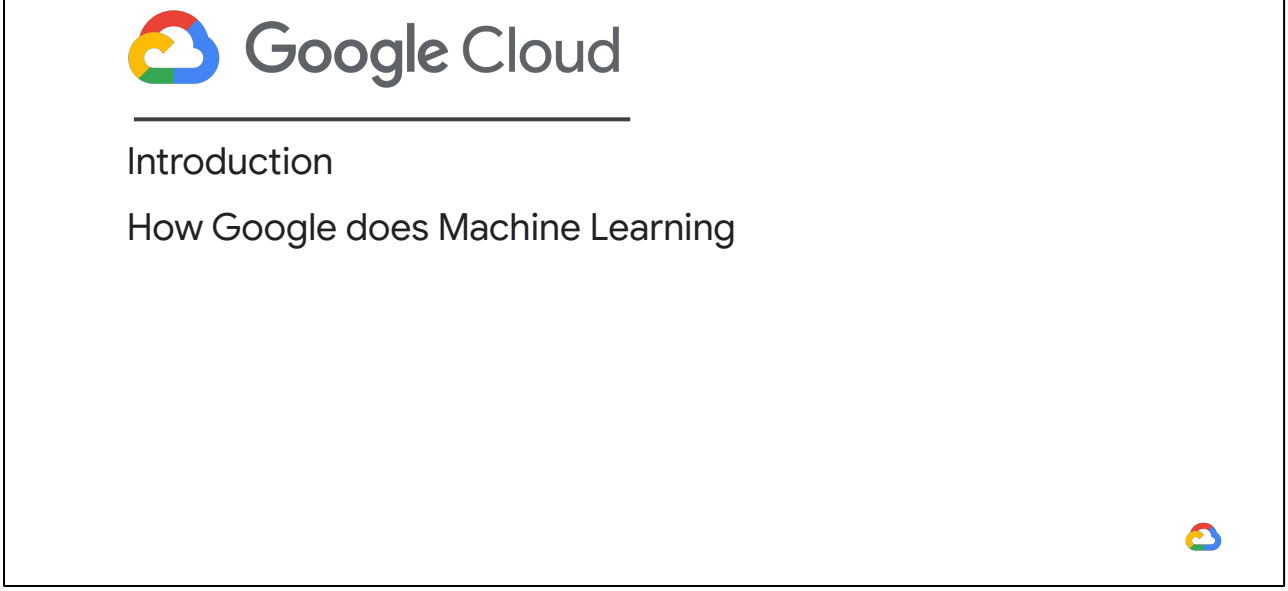
Google Cloud

Day 1



Welcome to How Google does Machine Learning.

Machine Learning with TensorFlow on GCP

How Google does Machine Learning

Launching into ML

Introduction to TensorFlow

Feature Engineering

The Art and Science of ML

This is the first chapter of the *Machine Learning with TensorFlow on GC* specialization.

Learn how to...

Convert a candidate use case to be driven by machine learning

Recognize biases that machine learning can amplify

Leverage Google Cloud tools and environment to do machine learning

Gain a broad perspective on machine learning and where it can be used

Frame a business use case as a machine learning problem

Agenda

Introduction

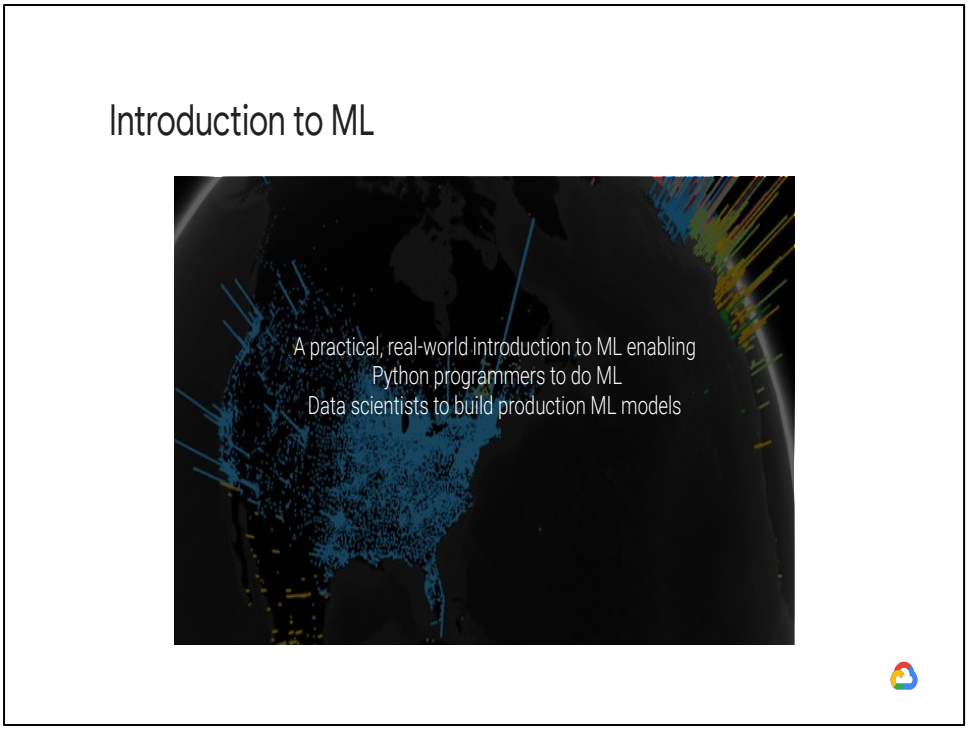
What it Means to be AI-first

How Google does ML

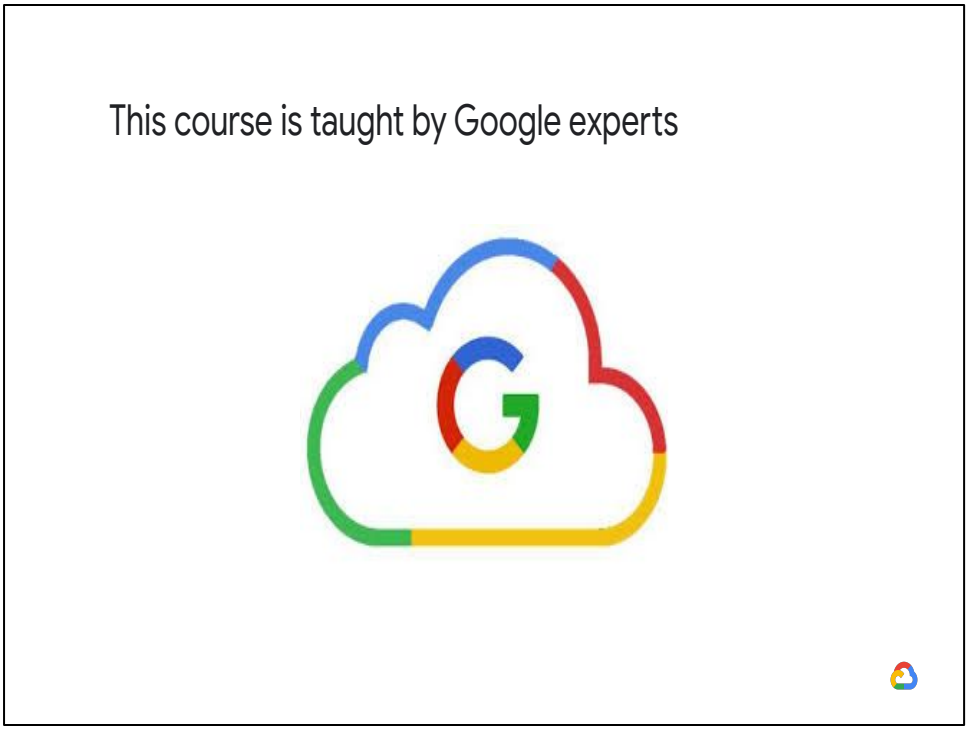
Inclusive ML

Python Notebooks in the Cloud

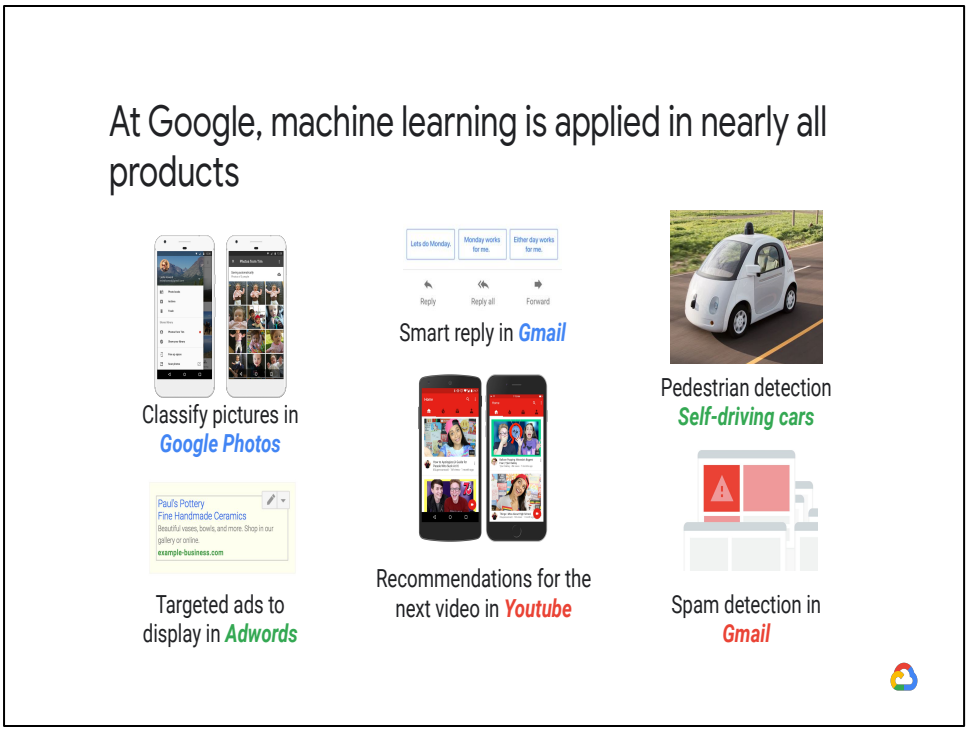
Summary



The aim of the specialization is to give you a practical, real-world introduction to machine learning. The goal is to enable you, whether you are a Python programmer or a data scientist, to do machine learning and build production machine learning models.
What will you learn?



This course, on how to do machine learning, is taught by Googlers who work in ML … So, why would you look to Google and Googlers to learn about ML?



At Google, you apply ML in pretty much all of your products.
You're learning this course so that you can share some of what you've learned, and you're super excited to see what you build based on what you learn in this set of courses! Of course, hopefully you build on Google Cloud.

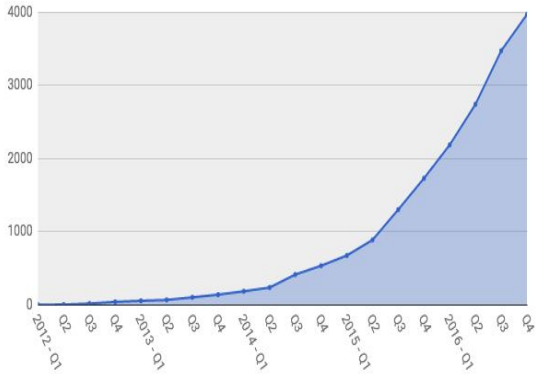
Machine learning has completely transformed the way *we (Google, that is)* do business, including:
-- how we approach new and existing products
-- how we design our products, and
-- how we approach new challenges.

Many of you probably interact with Google services (Photos, Youtube, Gmail, Inbox) every day.
All those services benefit from, and in some cases depend upon, ML.
We keep track of how much we use machine learning, deep learning, in particular.
Over the last few years, we have produced over 4000 TensorFlow ML models.

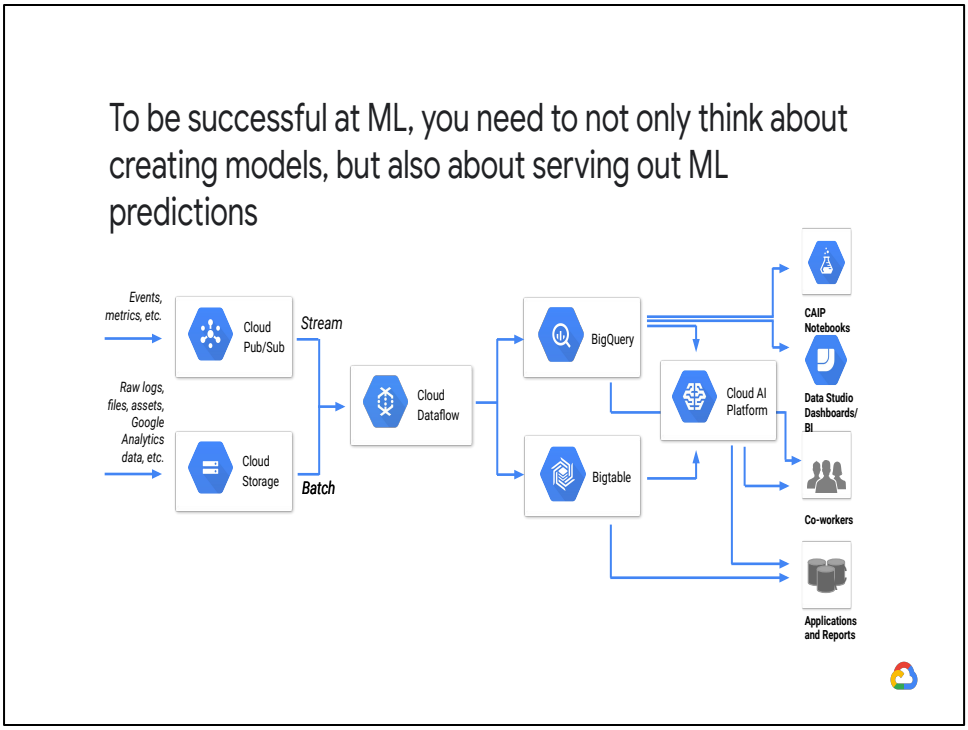
Image (Google Photos):
https://www.blog.google/products/photos/google-photos-500-million-new-sharing/
Image (Spam Detection):
https://www.google.com/search/howsearchworks/mission/site-owners/
Image (Self Driving Car):
https://www.blog.google/topics/alphabet/self-driving-vehicle-prototypes-on-road/
Image (youtube):

https://youtube.googleblog.com/2016/04/a-new-mobile-design-for-your-home.html
Image (adwords): https://adwords.google.com/home/how-it-works/
Image (Smart Reply):
https://www.blog.google/products/gmail/save-time-with-smart-reply-in-gmail/

There are over 4000 TensorFlow machine learning models in production at Google, and it has transformed our company



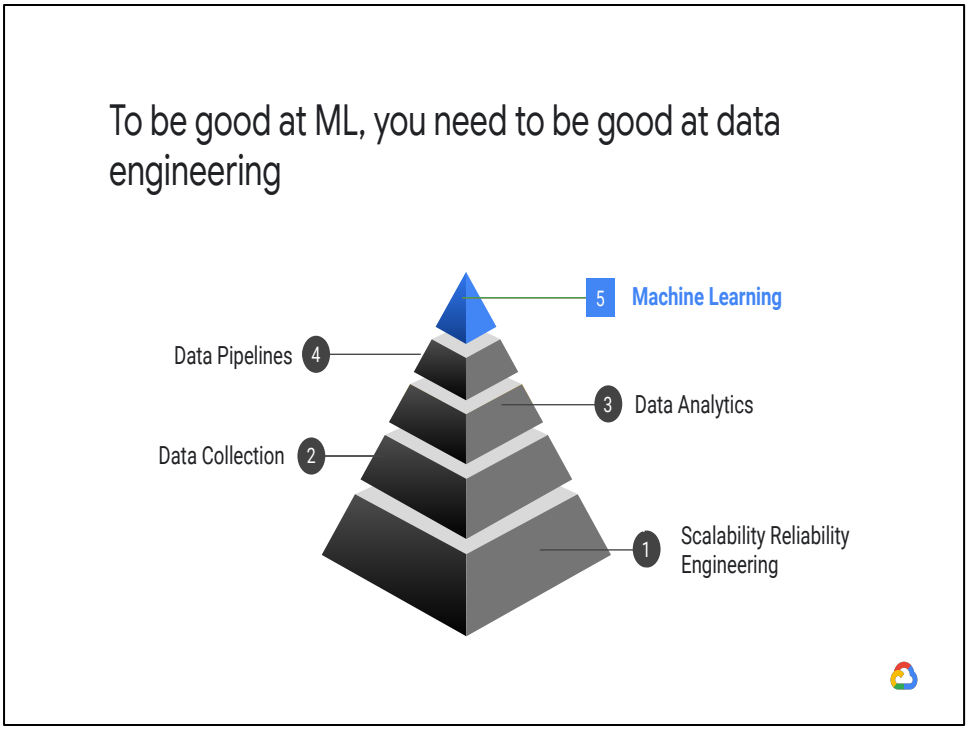
You heard that right. 4000. There are over 4000 TensorFlow machine learning models in production at Google.
In 2012, that number was near zero; and by the end of 2016, we were crossing 4000.
In this specialization, you will learn to use the
technology we use; this course covers how our engineers infuse ML into our products.



One of the key lessons we have learned along the way is that it is important to think about ML serving, not just about ML training.
When you say machine learning to most people, they think about the complex pipeline on the left of this diagram. It is certainly where
you, as a data engineer or data scientist, will spend a lot of your time.

However, the key reason you are doing machine learning is on the right-hand side of this diagram -- you want to serve out those predictions to decision makers using notebooks, dashboards, applications and reports.

Operationalizing an ML model is *hard* and many projects fail to make it to the predictions stage.
One of the lessons we at Google learned was that, in order to reduce our chance of failure, we should make sure that we process batch data and streaming data the same way. Cloud Dataflow in this diagram … it is open-sourced as Apache Beam … Dataflow helps us treat batch and stream the same way.

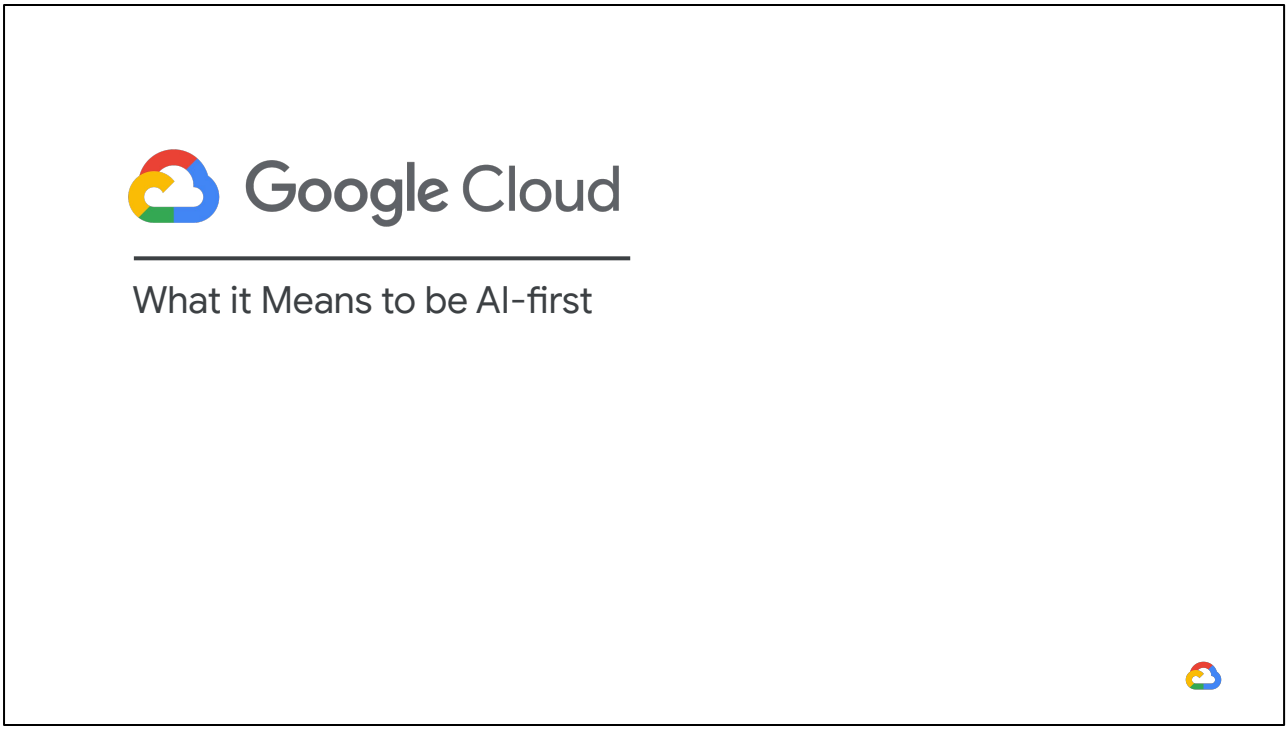


Cloud Dataflow is just one example of how, on Google Cloud, you get to take advantage of our experience in building ML infrastructure.

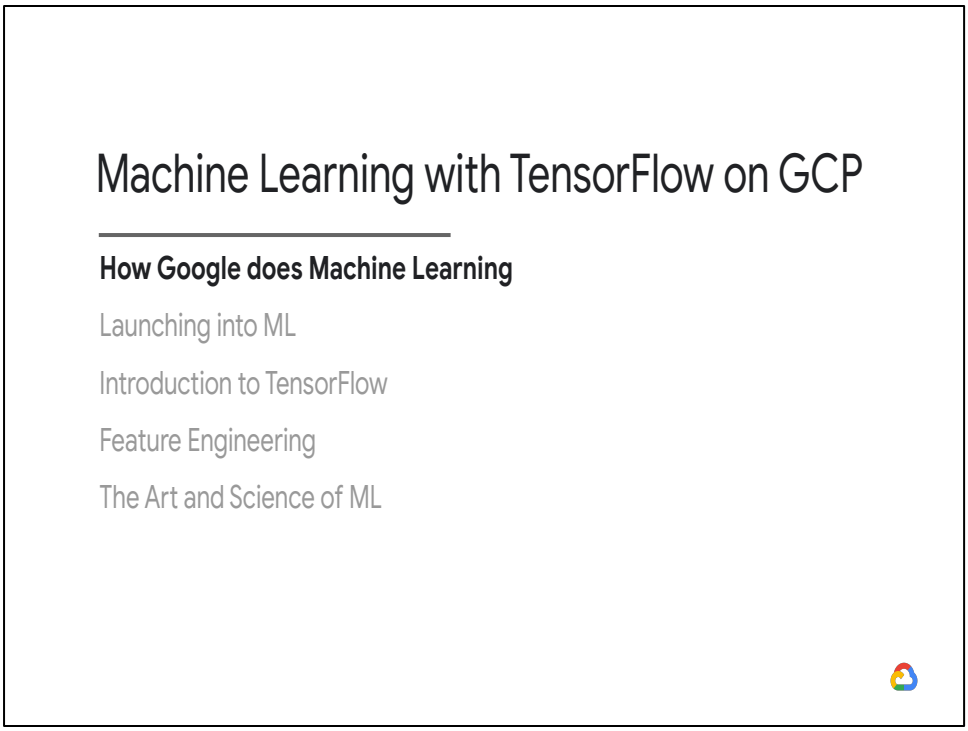
If you haven't taken our Data Engineering specialization on Coursera, I strongly encourage you to take it. But in this track, we will cover the key pieces as we go along. Fortunately, for those of you data scientists out there, data engineering is not hard to learn. On GCP, the key services are all serverless and they are all managed infrastructure. In this course, we will show you how to build batch and streaming data pipelines. We won't talk much about data collection and data analytics, though.

By building your data pipelines on Google Cloud, you essentially get to take advantage of the scalability, reliability and sheer engineering prowess that Google brings to running ML systems.

[end-video]



Welcome to ‘What it means to be AI-first.’



This is the first module in the first chapter of the *Machine Learning with TensorFlow on GCP* specialization.

Machine learning has completely transformed the way Google builds its products. In this module, you will learn what is meant by the Google strategy “to be AI-first”, and what that means in practice.

The goal is not to just talk about Google, but about how you can apply what you have learned in your journey to using ML everywhere. You will learn how to take a business problem and look at it with fresh insight. This insight will help you employ ML to help augment your users’ decision-making.

A few years ago, Google decided to focus on mobile, and began to do things “mobile-first”. Instead of building applications for the web, for example, and then porting them to mobile devices, Google decided it would build for mobile first. This sort of slogan serves as a rallying cry across the organization, and helps focus a collective mind on new opportunities.

It is in that context that you should understand “AI-first”. What this means in practice is quite different from what “mobile-first” meant in practice. In the case of AI-first, it means that Google consciously rethinks the approach to how products are built.

Learn how to...

Build a data strategy around ML

Identify and solve ML problems

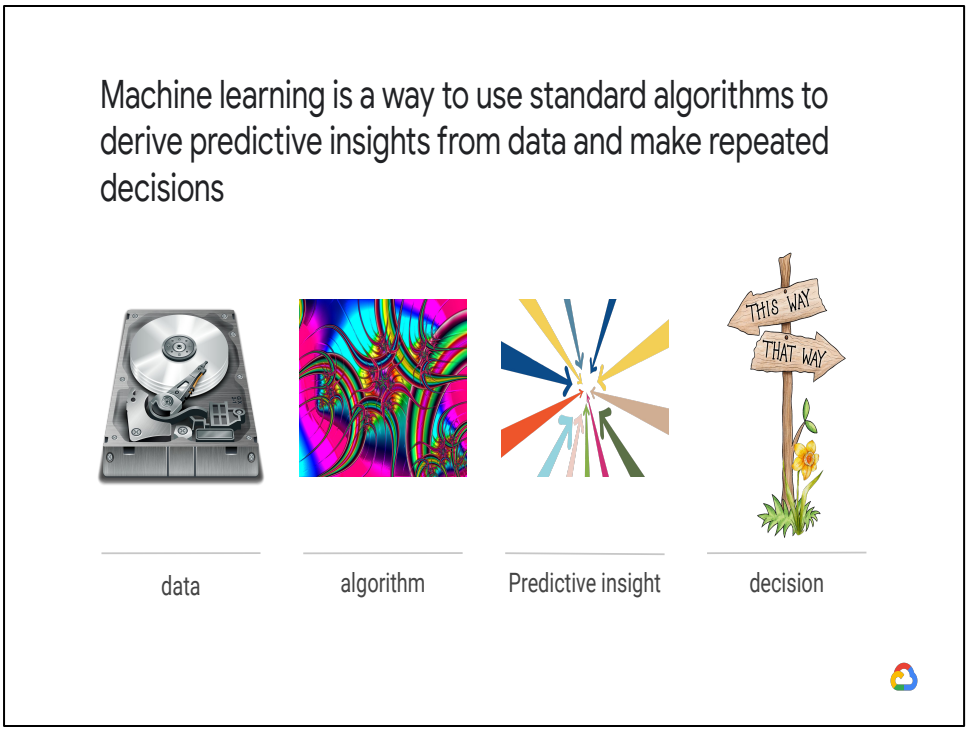
Infuse applications with ML

Use ML creatively, to delight your users

In this module, you will build a data strategy around ML, learn how to identify and solve ML problems, and use ML creatively.

You will start by learning what is meant by the term machine learning, and what it means to be AI-first. This comes down to a unique answer to the question: What kinds of problems can ML solve?

You will then be introduced to three specific strategic things you will do: infusing applications with ML; building a data strategy; and finally, using ML creatively.



Machine learning is a way to derive ‘predictive’ insights from data. You do this using algorithms that are relatively general and applicable to a wide variety of datasets.

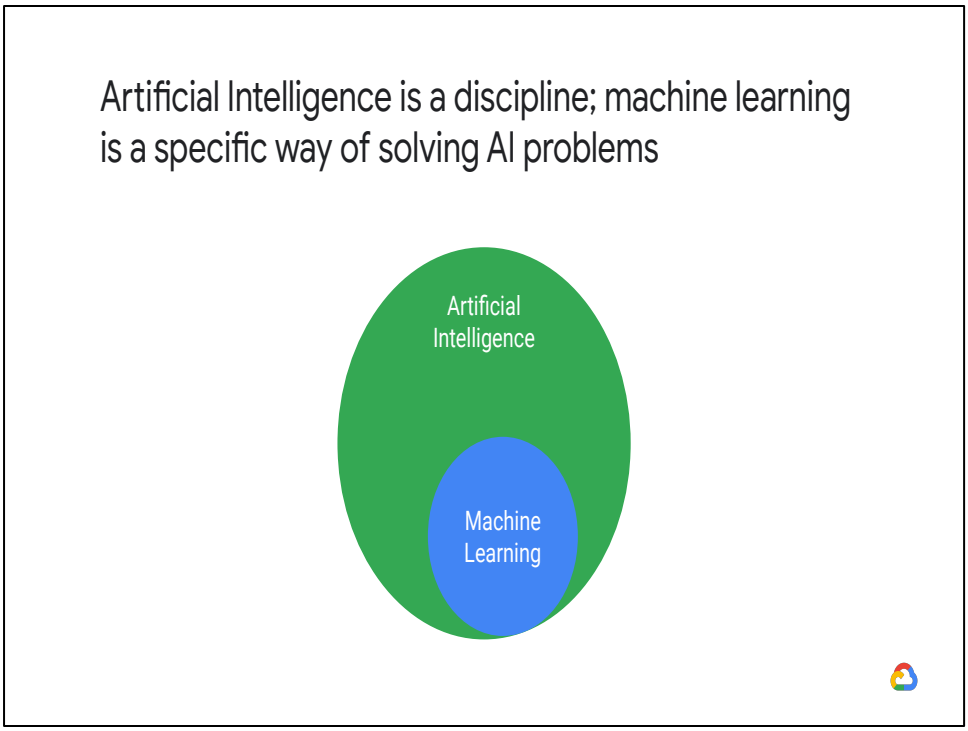
Think back to your company. How do you use data today? Perhaps you have a dashboard that business analysts and decision-makers view on a daily basis? Perhaps a report that they read on a monthly basis? That’s an example of backward-looking use of data -- looking at historical data to create reports and dashboards. This is what people tend to mean when they talk about business intelligence. A lot of data analytics is backwards-looking.

Of course, the point of looking at historical data might be to make decisions. Perhaps business analysts examine the data and suggest new policies or rules? They suggest, for example, that it might be possible to raise the price of a product in a certain region. Now, the business analyst is making a predictive insight, but is that scalable? Can the business analyst make such a decision for every product in every region? Can they dynamically adjust the price every second? In order to make decisions around predictive insights repeatable, you need machine learning. You need a computer program deriving such insights.

So, machine learning is about making many predictive decisions from data. It’s about scaling up business intelligence and decision making.

https://pixabay.com/en/hard-disk-storage-computer-159264/ (cc0)
https://pixabay.com/en/fractal-pattern-abstract-form-142748/ (cc0)

https://pixabay.com/en/arrows-inside-pressure-request-2029157/ (cc0)
https://pixabay.com/en/sign-direction-kids-cute-paint-2792576/ (cc0)

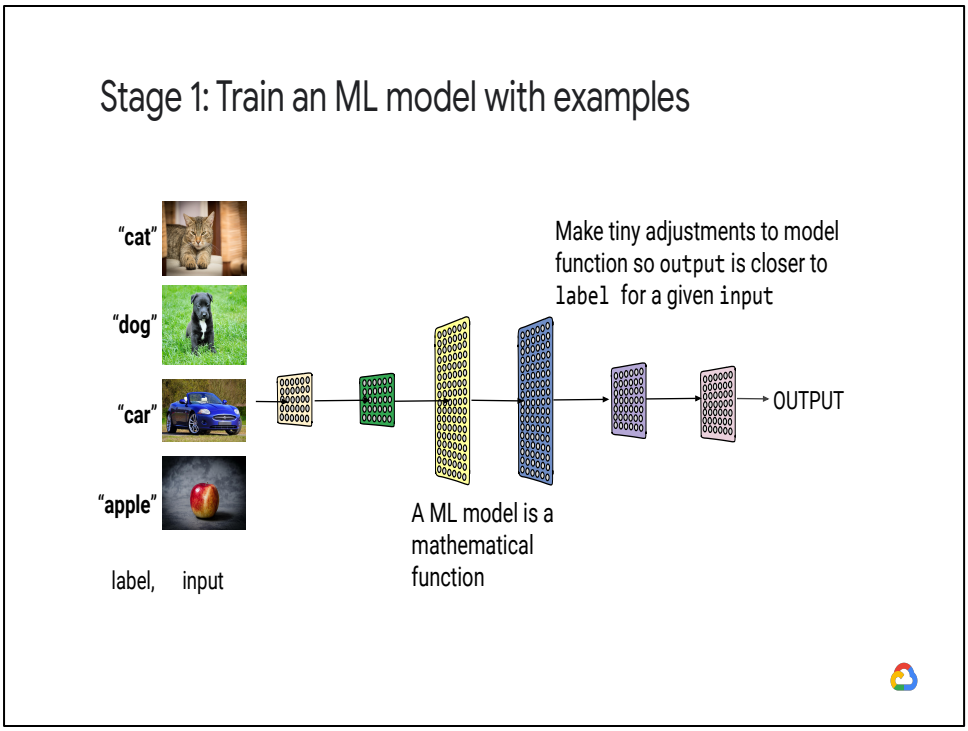


A very common question asked is: What's the difference between AI and machine learning?

One way to think about it is that AI is a discipline, like physics. AI has to do with the theory and methods to build machines that think and act like humans.

Machine learning is a toolset, like Newton's laws of mechanics. Just as you can use Newton's laws to figure out how long it will take a ball to fall to the ground if you drop it off a cliff, you can use machine learning to solve certain kinds of AI problems.

The basic difference between ML and other techniques in AI (for example, expert systems) is that in ML, machines learn. They don't start out intelligent, they become intelligent.



The first stage of ML is to train an ML model with examples. The form of ML that will be focused on is called supervised learning, and in supervised learning, you start from examples.

An example consists of a label and an input. For example, suppose you want to train an ML model to look at images and identify what is in those images. The true answer is called the label, so “cat” for the first image and “dog” for the second image. The image itself, the pixels of the image, are the input to the model.

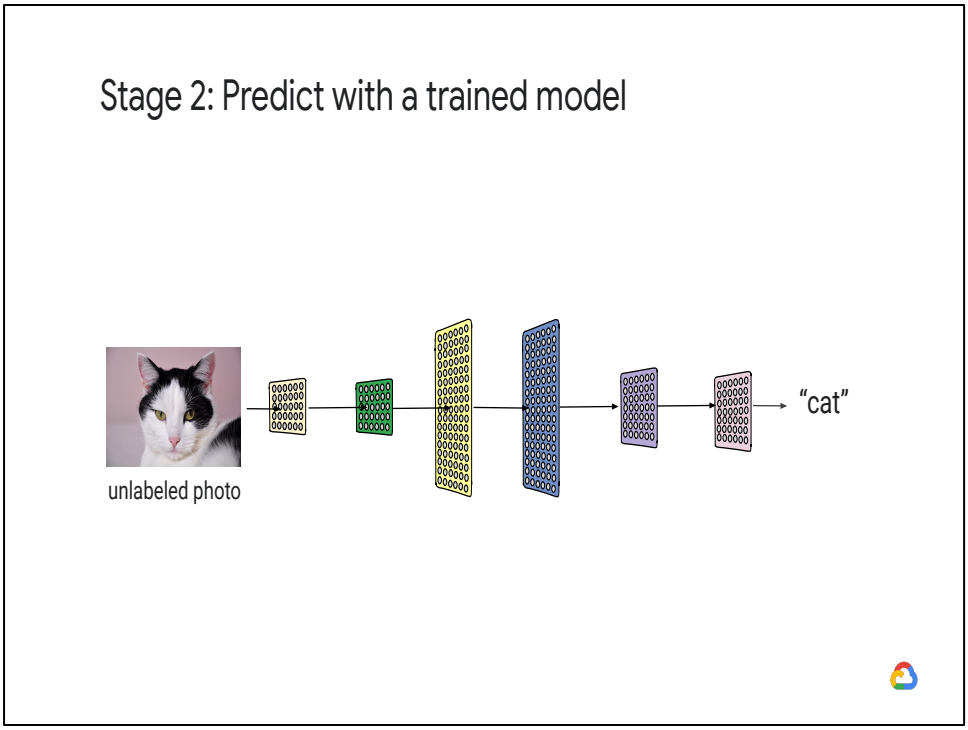
The model itself is a mathematical function of a form that can be applied to a wide variety of problems.

The models used in ML have a bunch of adjustable parameters. When you “train” the model, you make tiny adjustments to the model so that the output of the model -- the output of the mathematical function -- is as close as possible to the true answer for a given input. Of course, you don’t do this on one image at a time. The idea is to adjust the mathematical function so that overall, the outputs of the model for the set of training inputs is as close as possible to the training labels.

The key thing is that ML, at least ML of the form considered in this course, which is the most mature form of ML, relies on having a dataset of labeled examples. Labeled examples are input + the true answer.

https://pixabay.com/en/domestic-cat-cat-adidas-relaxed-726989/ (cc0)

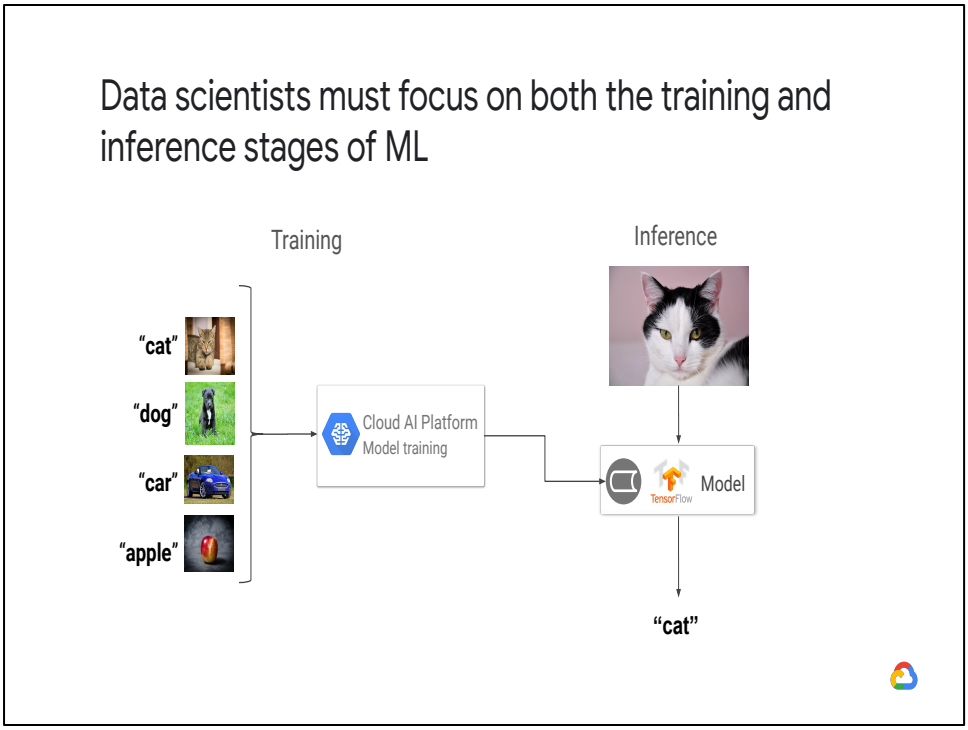
https://pixabay.com/en/dog-young-dog-puppy-280332/ (cc0)
https://pixabay.com/en/sports-car-vehicle-transportation-1317645/ (cc0)
https://pixabay.com/en/apple-education-school-knowledge-256268/ (cc0)



After the model is trained, you can use it to “predict” the label of images that it has never seen before.

In this example, you are inputting this image to the trained model. Because the network has been trained, it is correctly able to output “cat”. Note that the cat image on this slide is different from the one before it. It still works because the ML model has “generalized” from the specific examples of cat images we showed it, to a more general idea of what a cat is and what it looks like.
The key to making an ML model generalize, is data - and lots and lots of it. Having labeled data is a precondition for successful ML.

https://pixabay.com/en/cats-a-normal-cat-pet-796437/ (cc0)

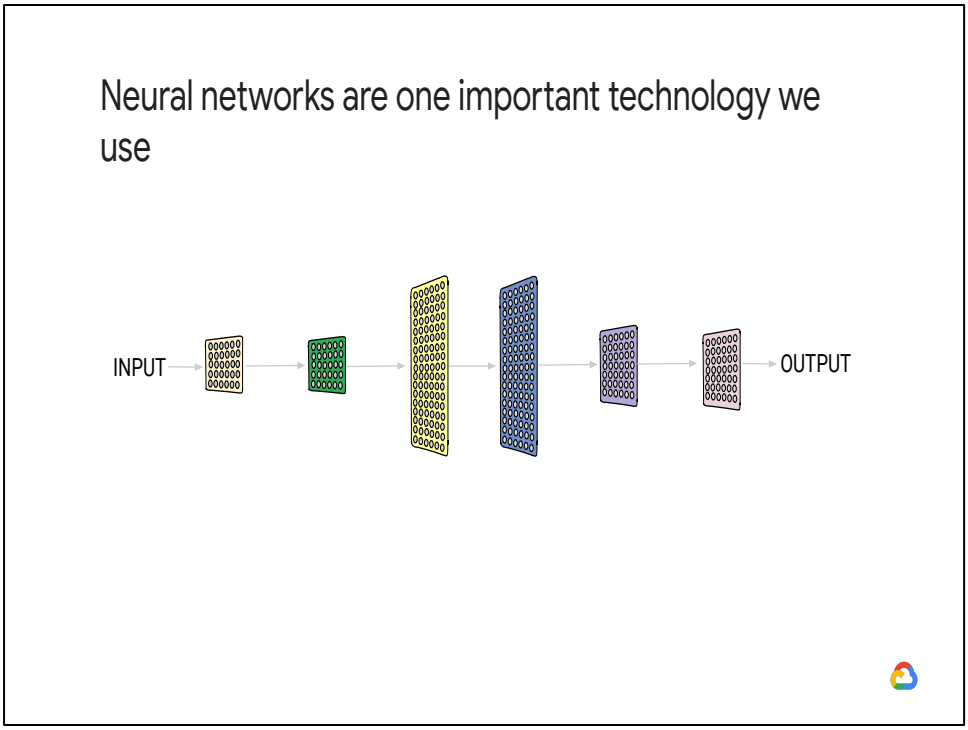


It's important to realize that ML has two stages: training and inference. Sometimes, people refer to "prediction" as "inference" because "prediction" seems to imply a future state. In the case of images like this, we are not really "predicting" that it's a cat, just "inferring" that it's a cat based on the pixel data.

It can be tempting, as a data scientist, to focus all your energy on the first stage. But this is'nt enough. You need to be able to operationalize the model -- put the model into production so that you can run inferences.

Many books on ML, blog posts, and university courses tend to ignore this second stage of ML. But in the real-world, what's the use of training an ML model if you can't use it? In this specialization, you'll be shown ML end-to-end, where end-to-end means putting ML models into production.

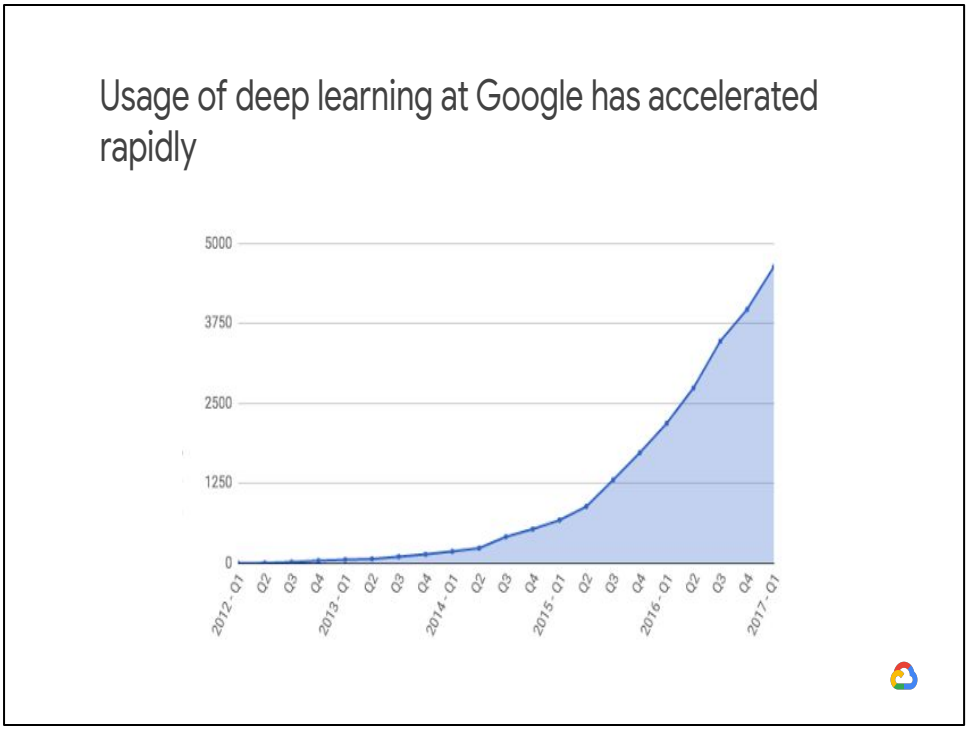
https://pixabay.com/en/cat-cute-cat-baby-kitten-pet-1992140/(cc0)
https://pixabay.com/en/mouse-rodent-cute-mammal-nager-1708347/



In the previous slides, you drew the mathematical model in a specific form. The model consists of many layers, arranged one after the other. The input passes through the first layer, then the second, etc. with each of the layers being a simple mathematical function. The entire model therefore consists of a function of a function of a function, and so on.

The diagram shows a mathematical model called a neural network. There are other common mathematical models used in ML: linear methods, decision trees, radial basis functions, ensembles of trees, radial basis functions followed by linear methods, the list goes on. But the discussion is about neural networks.

Traditionally, neural network models didn't have this many layers. Neural networks date back to the 1970s, but they used to have only one hidden layer. The reason had firstly to do with computational power: training deep neural networks, neural networks with lots of layers, takes a lot of computing power. Secondly, availability of data. As you add more layers, there are more and more weights to adjust, so you need lots and lots more data. Thirdly, computational tricks. It turns out that if you just add layers, you'll run into some issues. The neural networks will take a long time to train. Some of the layers will become all zero. Or they'll blow up and become all NaN (not a number). So, the research community need to develop a number of tricks and techniques to get deep neural networks to work.

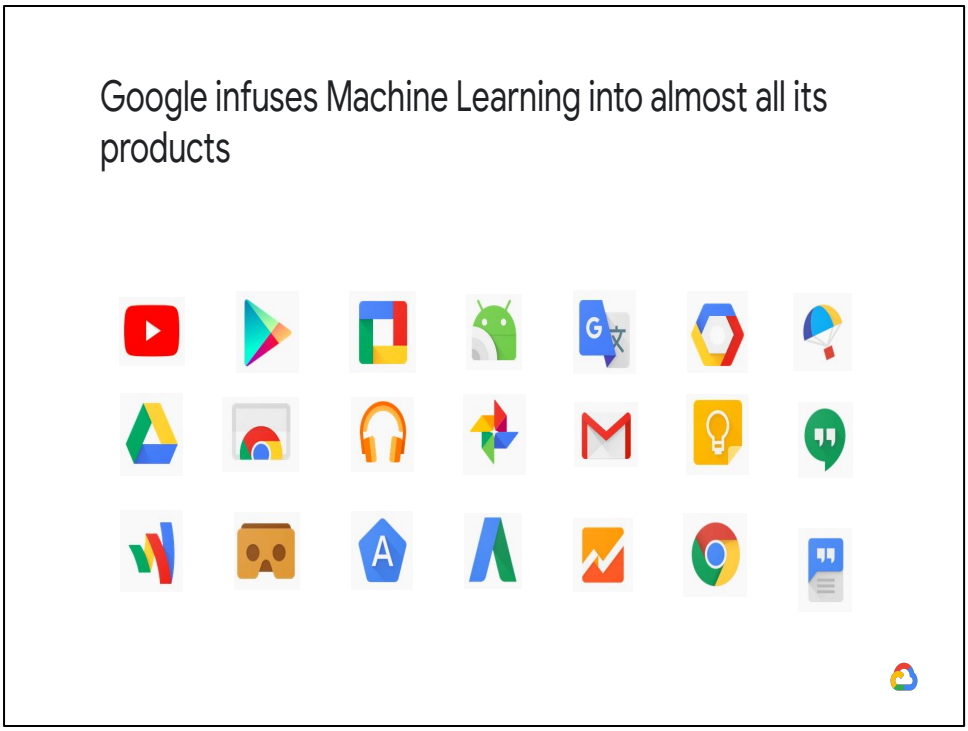


In the last few years, neural networks have proven themselves to be the best, or near-best, in a wide variety of tasks. Even tasks that used to be thought to be unsolvable with ML. Neural networks have enabled dramatic improvements in really hard ML problems, like language translation, image classification, speech understanding, etc. And they work just as well, or better, on structured data problems as traditional ML methods such as support vector machines or boosted/bagged decision trees.

You can see this at Google. The use of deep learning at Google has accelerated rapidly. There were pretty much no deep learning models four years ago, and now Google has more than 4000 deep learning models.

You will use neural networks exclusively in this specialization. You'll start off on structured data problems, and once you know how to build an end-to-end pipeline, you'll take that knowledge and show how to do image problems, and sequence problems, and recommendation systems.

But, look again at this graph. 4000+ models? How can there be so many ML models?

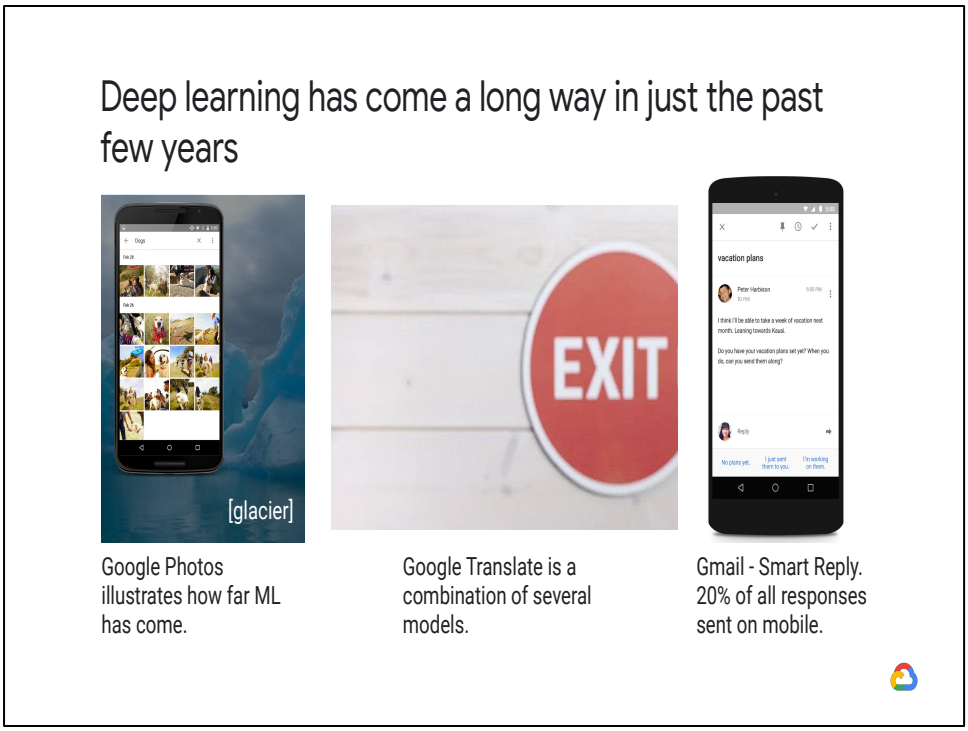


ML is part of pretty much every Google product out there. Whether it's YouTube, or Play, or Chrome, or Gmail, or Hangout, they all use ML. It's not that there's just one ML model at YouTube. There are dozens of ML models per product.

This is something that takes getting used to. You might look at a business problem, say, how to forecast whether an item will go out of stock, and think of it as a single ML model that you have to build. In practice, you'll have to build many ML models to solve this problem. You might have to break this problem down into smaller problems based on your knowledge of the business.

For example, your first model might be to predict demand for the product at the store location. Your second model might predict the inventory of this item at your supplier's warehouses and nearby stores. You might need a third model to predict how long it's going to take them to stock your product and use this to predict which supplier you'll ask to refill the shelf, and when. Of course, all these models themselves might be more complex. The model to predict the demand for milk is going to be very different from the model to predict the demand for dry noodles. The model for restocking electronics is very different from the model for restocking furniture.

There isn't one ML model; there are dozens of ML models per product. For this training, you'll be shown how to train, deploy, and predict with a single model. In practice though, you'll be building many ML models to solve a use case. Avoid the trap of thinking of building monolithic, one-model-solves-the-whole-problem, solutions.



Back to the Google experience. You've probably used one or more of our 4000 models.

Deep learning has come a long way in just the past few years, and nothing illustrates that better than Google Photos. This is the Google product where you can upload photos from your camera to the cloud. You don't need to tag it -- the ML software tags images for you, so that you can then find images.

The Google Translate app lets you point a phone camera at a street sign and it translates the sign for you. This is a good example of a combination of several models that's quite intuitive.
One model to find the sign.
Another model to read the sign (do OCR -- optical character recognition -- on it).
A third model to translate the sign. (Maybe a third model to detect the language, and a fourth model to translate the sign).
A fifth model to superimpose translated text.
Perhaps even a sixth model to select the font to use, and so on.

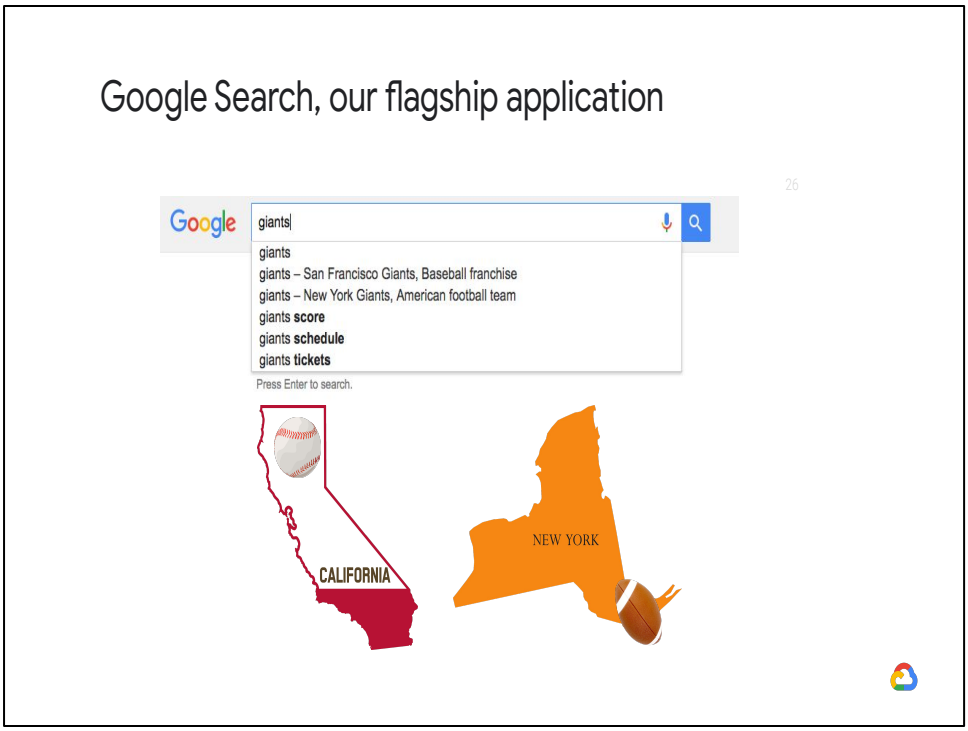
Smart Reply is the feature of Inbox and Gmail where the email program suggests three possible responses to received emails. This is arguably the most sophisticated ML model in production today.
Why do you think that is? It's a sequence-to-sequence model. It takes the received email as input and generates the response as the output. Text, is usually thought of a sequence of words.

The ML model here needs to understand a small body of text, and predict three dissimilar answers. You'll be introduced to sequence models later.

Source: go/srquality

In the previous topic, ML was discussed as being a way to derive repeated predictive insights from data. Then, the two stages of ML were explored: a training phase where you teach the algorithm using labeled examples, and a prediction or inference stage where you use the trained model to make inferences on new data. Lastly, you were shown a few examples of ML in action -- Photos, Translate, Smart Reply, all from Google products.

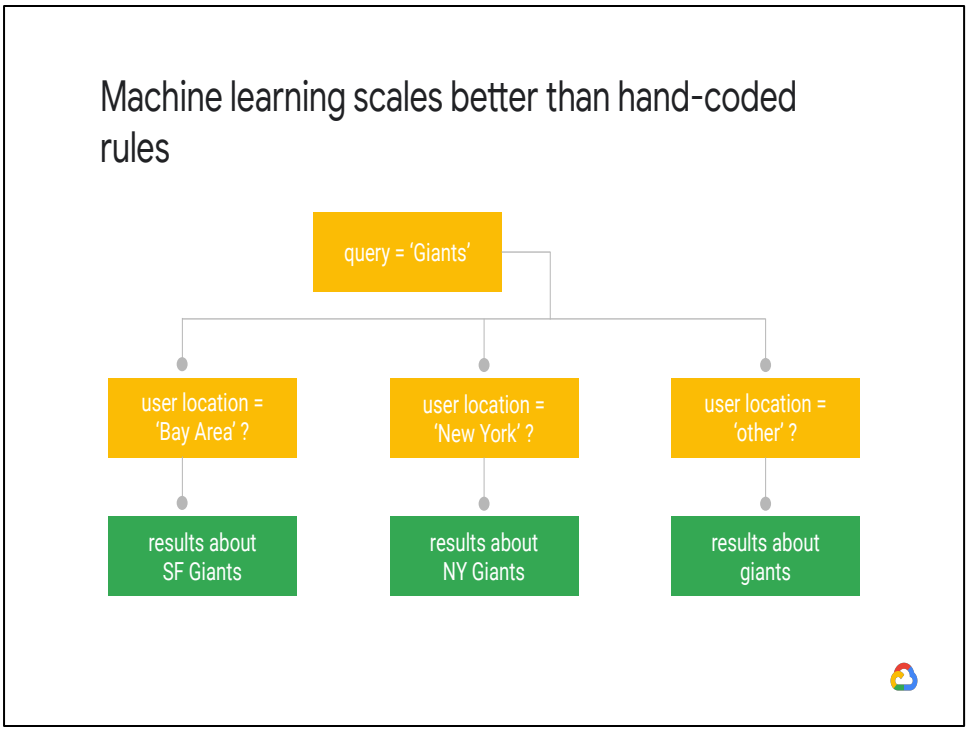
So how do you get to the point where your company is innovating like this in ML? Our execs had a unique answer to the question of: "What kinds of problems can machine learning solve?"



Consider Search. This is, of course, the flagship application at Google.

If you type in “giants”, should we show you San Francisco giants or New York giants? How would you do it?

https://pixabay.com/en/baseball-ball-sport-team-batting-25761/ (cc0)
https://pixabay.com/en/california-usa-americana-2217654/ (cc0)
https://pixabay.com/en/map-new-york-state-geography-43774/ (cc0)

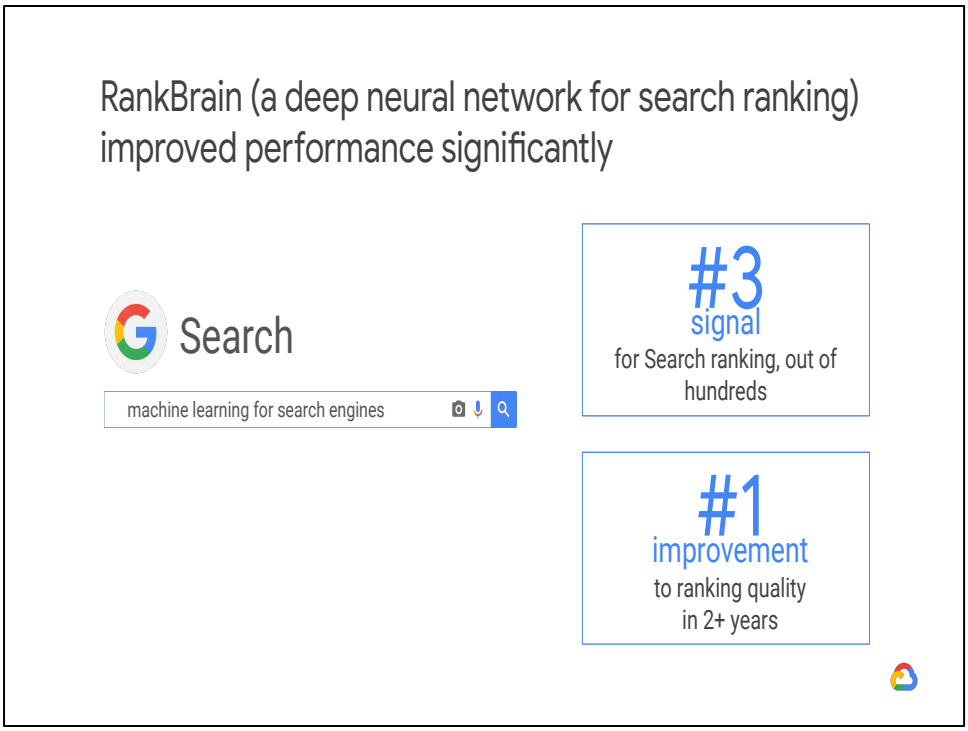


A few years ago, this is how Google Search worked.

There were a bunch of rules that were part of the search engine codebase to decide which sports team to show a user.
If the query is 'giants' and the user is in the Bay area, show them results about San Francisco Giants.
If the user is in the New York area, show them results about New York Giants.
If they are anywhere else, show them results about tall people.

This is for just one query.
Multiply this by the large variety of queries that people make, and you can imagine how complex the whole codebase had become.
The codebase was getting unwieldy. Hand-coded rules are hard to maintain.

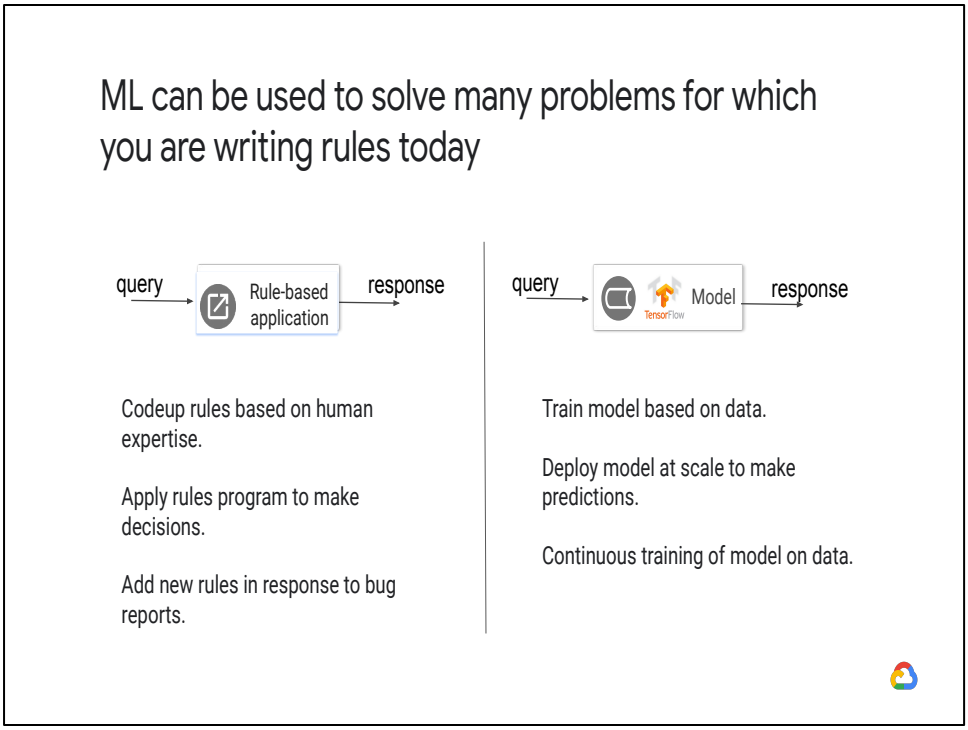
Why not try machine learning? ML scales better because it's all automated. Google knew when they showed people results, which of the links they actually clicked on.
How about training a ML model so that it could do the search ranking?



That was the essential idea behind RankBrain, a deep neural network for search ranking. It outperformed many human-built signals. You could replace many of the hand-coded rules with ML.

The neural network ended up improving Google search quality dramatically. Plus, the system could continually improve itself based on what users actually preferred.

Replacing heuristic rules by ML - that's what ML is about.



What kinds of problems can you solve with ML? Anything for which you are writing rules today. It's not just about predictive analytics! Google Search is not a predictive analytics application, but you use ML for it. Note that this is a far more expansive answer to "what kinds of problems can ML solve?" That's what is meant when Google say it's an AI-first company. Google think of ML as the way to scale, to automate, to personalize. Think about all the heuristic rules you are coding up today; provided you can collect the right data, you may be able to do it using ML.

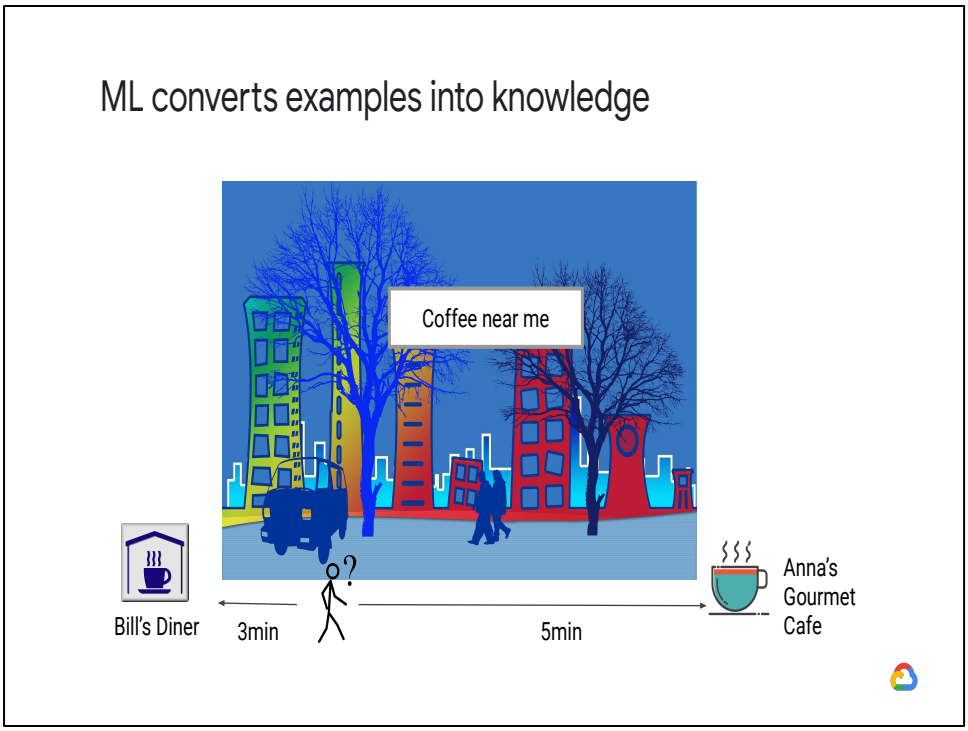
The way you think about problems changes when you do this. You don't think about coding up rules; you think about training models based on data. You don't think about fixing bug reports by adding new rules; you think in terms of continuously training the model as you get new data. And when you think of applying rules to inputs, you think in terms of deploying models at scale to make predictions.

So, how does this idea change the way you approach new problems? A few years ago, you found that certain types of queries were becoming more common. Japanese toys in San Francisco, live lobster in Kissimmee, vegan donuts near me.

These are hard queries, local queries. People are not looking for websites, but actually businesses on a map.

You could write rules for each of these, but it becomes unwieldy rather quickly. How do you approach it from an ML perspective? You start by thinking about how to collect the data to make it an ML problem.

Image (Lobster) cc0: https://pixabay.com/en/lobster-shediac-canada-2358898/
Image (Donuts) cc0: https://pixabay.com/en/candy-food-donuts-market-2696734/
Image (Beehive) cc0: https://pixabay.com/en/bees-combs-beekeeper-honeycomb-2368228/
Image (Toy) cc0: https://pixabay.com/en/panda-z-panda-toy-children-child-1299683/

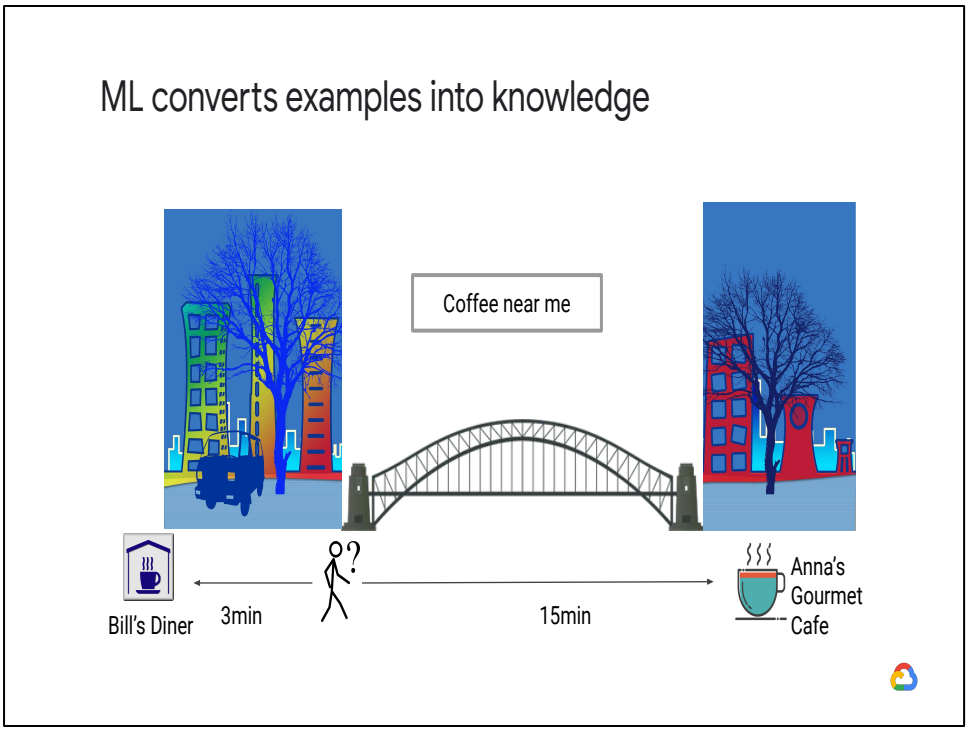


Here's an example. The query "Coffee near me".

The idea behind ML is to take a bunch of examples and convert that knowledge into future predictions. When you search for coffee near me, what are the examples? What is the future prediction?

The prediction end in this example is quite straightforward. There are two options: Bill's Diner carries coffee and it's only 3 minutes away. However, there's a gourmet coffee shop just 2 minutes more, and it's thought that you'd prefer the gourmet coffee to the sandwich shop.

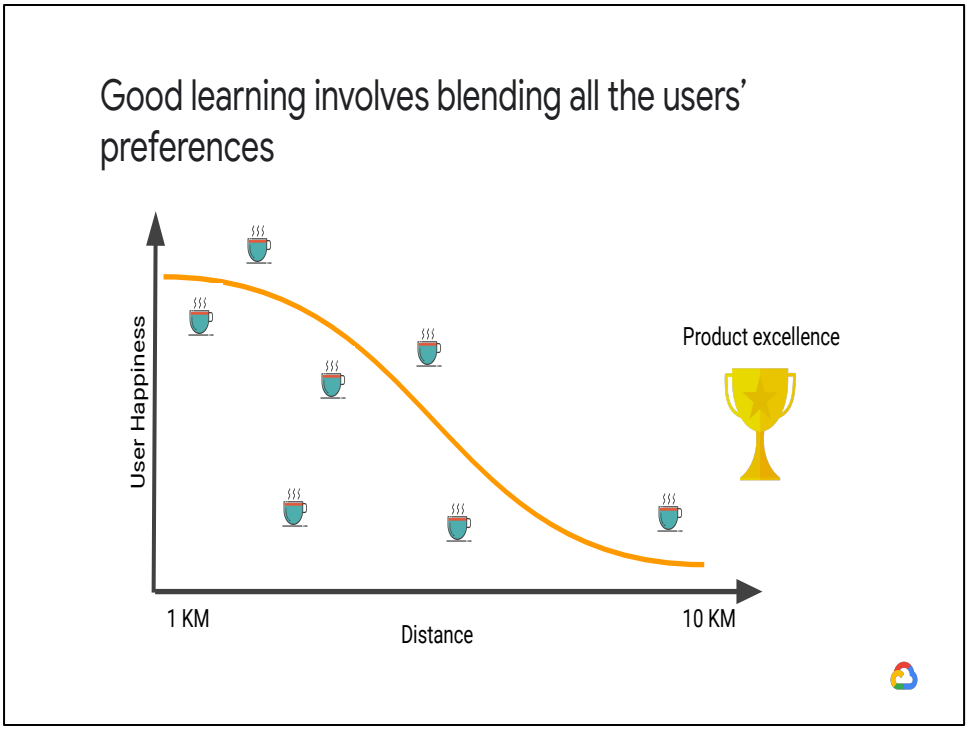
https://pixabay.com/en/city-trees-city-view-animated-107600/ (cc0)
https://pixabay.com/en/coffee-tea-symbol-cappuccino-hotel-32355/ (cc0)
https://pixabay.com/en/cup-icon-glass-symbol-design-flat-1849083/ (cc0)



On the other hand, if the gourmet coffee shop is across the bridge, it'll probably send you to the diner instead. Or, if the diner typically takes 10 minutes to serve coffee, or does not have take-away coffee (you have to sit down and eat), then perhaps the 15-minute walk is what you'd prefer.

How far is too far? How much does the rating of the restaurant, and the time it takes to serve, you matter? Rather than guessing and having a whole bunch of rules, we'd rather have users telling us.

https://pixabay.com/en/architecture-city-home-homes-107595/ (cc0)
https://pixabay.com/en/city-trees-city-view-animated-107600/ (cc0)



So, you look at a bunch of data and do a trade-off. Distance vs. quality of coffee, as well as other factors such as service time. In this example, just consider the distance. But where do you get this data?

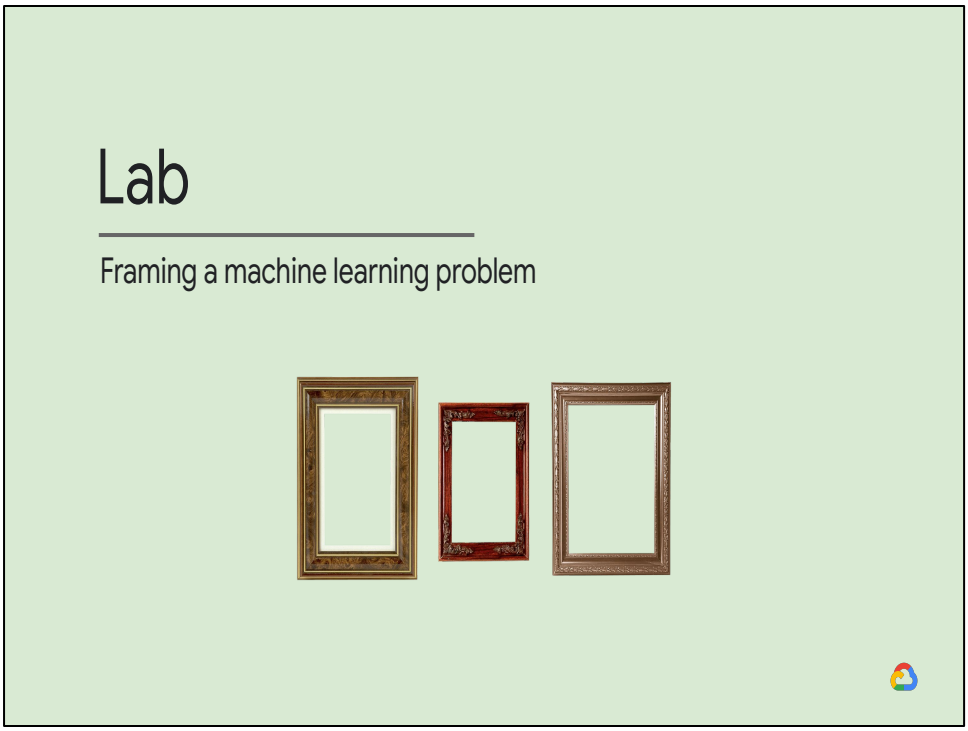
As an AI-first company, you might start with heuristics, but you do so with the mindset that you're going to throw away the heuristics just as soon as you've enough data about user preferences.

What you need are examples. Remember, example = labeled data. Here, the input = distance to shop, and label = does the user like the result or not?

Take an example of a shop 1 km away, and the user says: Great. I'll go 1km!
You ask another user 3km away, and they say: I don't even like gourmet coffee.
You aggregate a bunch of different examples, and eventually, when you realize when it's far enough away, nobody wants to go!

And then you try to fit your model.

Machine learning is about collecting the appropriate data, and then finding the right balance of good learning, trusting the examples.



In this lab activity, you'll choose 2 from a set of provided use cases and think about the problems from an ML perspective.

Pick two use cases that you are familiar with from the two slides that follow:
Cast this as an ML problem.
What is being predicted?
What data is needed?
Cast the ML problem as a software problem.
What is the API for the problem during prediction?
Who will use this service? How are they doing it today?
Now, cast it in the framework of a data problem.
What are some key actions to collect, analyze, predict, and react to the data/predictions (different input features might require different actions)

Image (Frames) cc0:
https://pixabay.com/en/frame-wooden-frame-decorative-2480747/

Cloud Machine Learning use cases

Manufacturing

- Predictive maintenance or condition monitoring
- Warranty reserve estimation
- Propensity to buy
- Demand forecasting
- Process optimization
- Telematics

Retail

- Predictive inventory planning
- Recommendation engines
- Upsell and cross-channel marketing
- Market segmentation and targeting
- Customer ROI and lifetime value

Healthcare and Life Sciences

- Alerts and diagnostics from real-time patient data
- Disease identification and risk satisfaction
- Patient triage optimization
- Proactive health management
- Healthcare provider sentiment analysis

This is the first set of potential use cases.

Cloud Machine Learning use cases (continued)

Travel and Hospitality

- Aircraft scheduling
- Dynamic pricing
- Social media – consumer feedback and interaction analysis
- Customer complaint resolution
- Traffic patterns and congestion management

Financial Services

- Risk analytics and regulation
- Customer Segmentation
- Cross-selling and up-selling
- Sales and marketing campaign management
- Credit worthiness evaluation

Energy, Feedstock and Utilities

- Power usage analytics
- Seismic data processing
- Carbon emissions and trading
- Customer-specific pricing
- Smart grid management
- Energy demand and supply optimization

This is the second set of potential use cases.

Example solution: Demand forecasting in manufacturing

ML problem:

What is being predicted?

How many units of widgets X should you manufacture this month?

What data is needed?

Historical data on # of units sold, price it was sold at, # of units returned, price of competitor product, # of units of all items that use widget X that were sold (e.g. if widget is a phone display panel, how many smartphones were sold, regardless of which display panel they carried?), economic figures (e.g. customer confidence, interest rate), this-month-last-year

Example solution: As a software problem

predictDemand(widgetID,
month=CurrentTime.month)

Who will use this service?

Product managers, logistics managers

How are they doing it today?

They examine trends of e.g. phone sales, overall economy, trade publications and make a decision

Example solution: As a data problem

Data problem:

Collect: economic data, competitor data, industry data, our figures

Analyze: craft features that our experts are looking at today from this data and use as inputs to model

React: automatic?

Agenda

What is ML?

What kinds of problems can it solve?

Infuse your apps with ML

Build a data strategy around ML

Use ML creatively, to delight your users

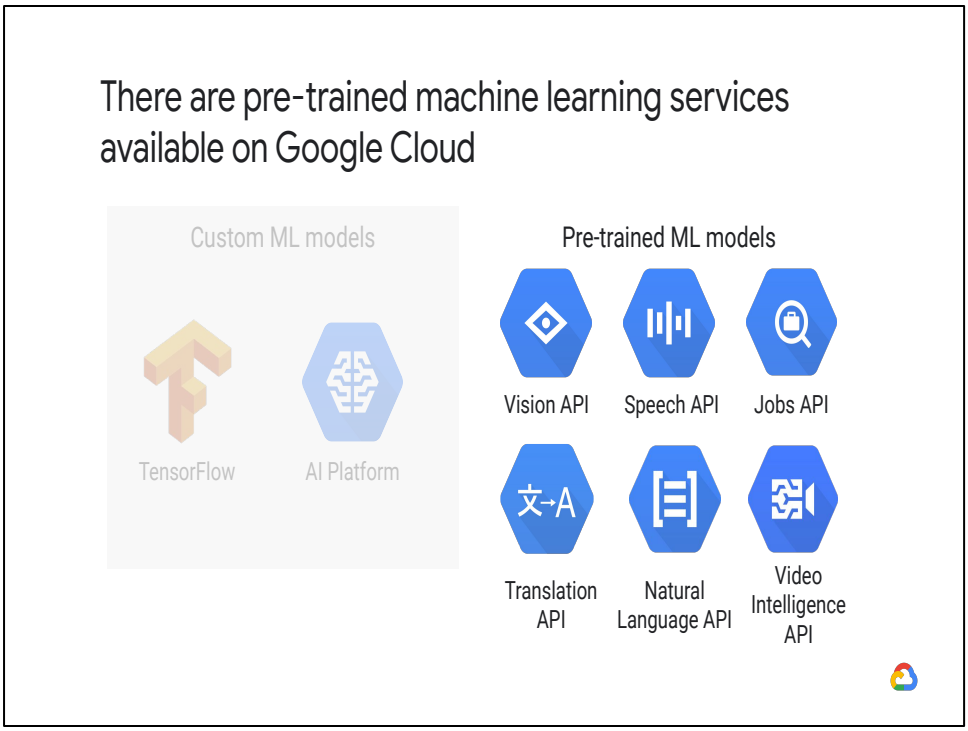
An easy way to add ML to your apps is to take advantage of pre-trained models. These are off-the-shelf solutions for cases where you don't need to build your own models.

Aucnet is the largest real-time car auction service in Japan, serving over 30 thousand dealers and running auctions worth nearly $4 billion dollars a year. Car dealers used to have to take multiple photos of each used car to sell, upload them to the service, and specify the model of the car and part of the car for every photo. It was a time consuming task for the dealers to do across thousands of photos every day.

Now, the new machine learning system can detect the model number of the car at high accuracy. It can also show the estimated price range for each model and recognize what part of the car is being photographed. With this system, the car dealers just drag-and-drop a bunch of unclassified photos, and check if the model and parts are classified by the system correctly.

A demo is available. Aucnet use case details: https://drive.google.com/open?id=1GPvRepg6ug-AnSkTtlG39fT8QVr6oY1UTYNumTsYDX8
Or, URL: https://konpeki.io/
Sample images: download this file (Use the images in "Selected for demo" folder)

It used to be that if you wanted to list a car, you needed to fill out a form. Boring! Aucnet said “upload images of your car” and we’ll tell you how much it’s worth. Fill out a form? Upload images from your camera? Which user experience is better?

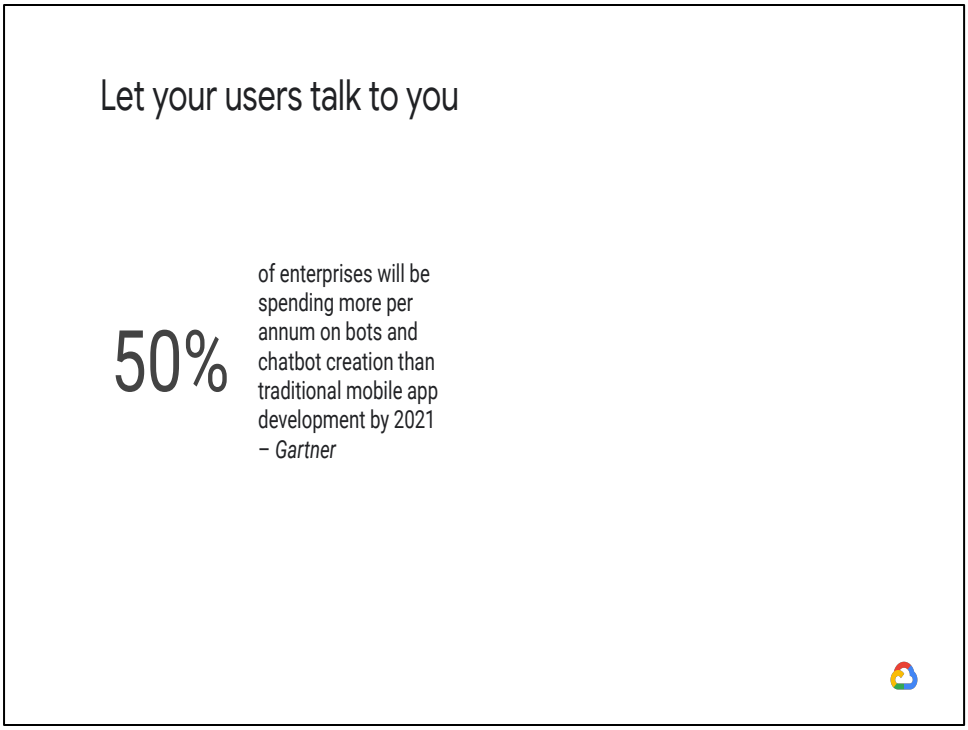


Aucnet built a custom image model on Google Cloud Platform using TensorFlow. These are on the left-hand side of this slide.

But increasingly, you don't have to do that. There are a variety of domains where Google exposes ML services trained with their own data. For example, if you want to transcribe speech, you could use the Speech API instead of having to collect the data, train it, and predict with it. There are many such pre-trained models. Such pretrained models are excellent ways to replace user input by ML.

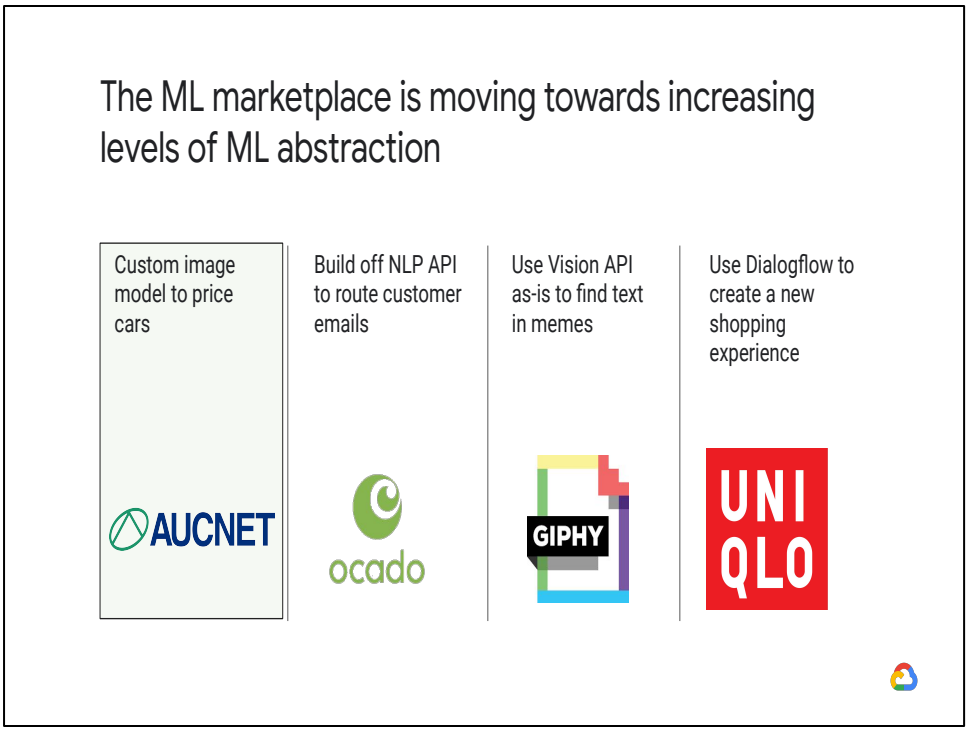
Another example of using a pre-trained model, Ocado is the world's largest online-only grocery, based in the UK. Customers send email, and traditionally each email would get read and routed to the appropriate department. That doesn't scale. So, Ocado turned to natural language processing. They were able to get sentiment, entities, and parsing syntax.

The computational technology helps Ocado parse through the body of emails, tagging and routing to help contact center reps determine the priority and context.



But increasingly, customers do not want to go to your website and click on a button. They don't want to send you an email. They want to talk to you. Interactively. To get their questions and concerns answered. Manually answering each call doesn't scale, and Gartner estimates that in a few years we'll be spending more on conversational interfaces than even on mobile apps!

Does this mean using the Speech API, transcribing it, and trying to make sense of it? No, what you're seeing here is a high-level conversational agent tool called Dialogflow.



Aucnet built their own custom model to classify car parts and estimate price.
Ocado used parsed results from the NLP API to route customer emails
Giphy uses the Vision API to find the text in memes using optical character recognition
The social media company used the Vision API to reject inappropriate uploads.
Uniqlo designed a shopping chatbot using Dialogflow.

In this specialization, we'll focus on building custom models. You'll learn how to do machine learning, and the level of abstraction you will work at will be around building custom models. But be aware that, increasingly, you'll get to incorporate machine learning into your applications primarily in the form of APIs.

Of course, someone will have to build these APIs for this marketplace, and perhaps you are the builder of such ML APIs.

Agenda

What is ML?

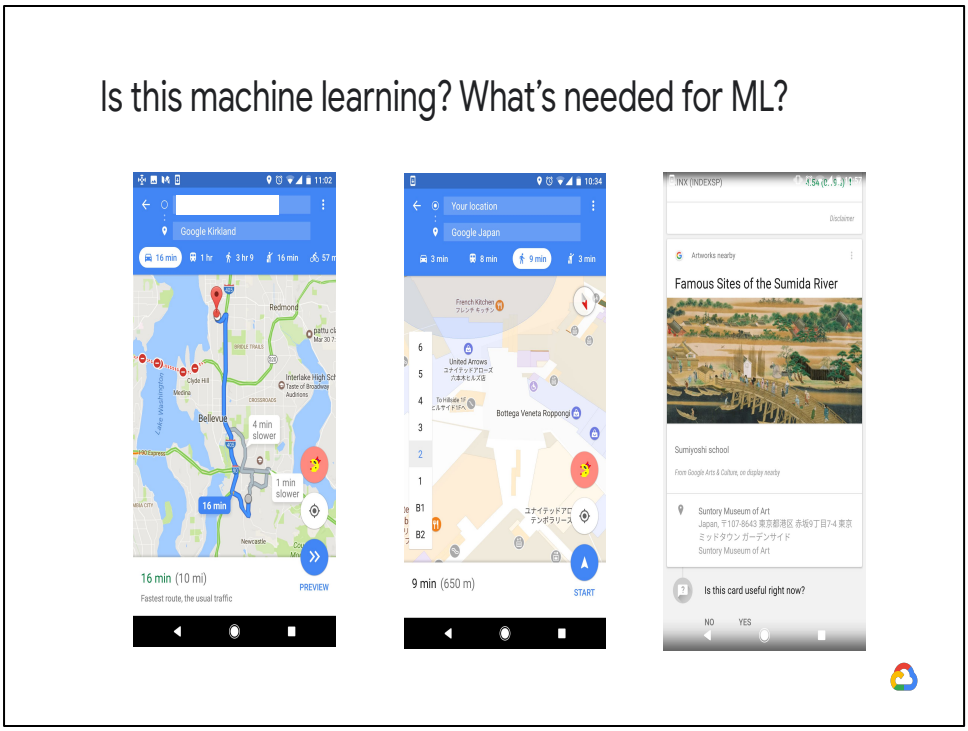
What kinds of problems can it solve?

Infuse your apps with ML

Build a data strategy around ML

Use ML creatively, to delight your users

Reimagine what data can help you do.



Google Maps can be used to illustrate several key points.
Take this map for example. You glance at your phone and it tells you the way to get to work. There are three possible routes, and today, the highlighted route is the fastest. Sometimes, you go to Google Seattle, crossing the floating bridge across Lake Washington and Maps tells you, helpfully, that the bridge is closed. Is this machine learning?
You could think of this as being just a set of rules. Google has to collect a lot of data to make this use case possible -- where the roads are, for one. The traffic on each road. Bridge closures.
The algorithm itself? Routing algorithms between A and B subject to a set of constraints? The A* algorithm is taught in undergraduate computer science classes. So not that complex once you have the data. This is the kind of thing you can do for countries at a time. Get data on the road network, and provide routing directions. Traffic and bridge closures are a little more difficult in that you have to work with a bunch of smaller government entities, but still not such a huge data problem. The logic, once you have the data, seems quite tractable.

Now, take the case in the middle. Still Maps. You're in Japan, making your way from your hotel to the Google office. You're in a subway station called Roppongi and Maps tells you that you're on floor #2.
How does it know? Whatever the data source is it uses -- wifi points, barometric pressure, typical walking speed, it's pretty obvious that this can not be a simple set of rules. The relevant data to train the model, and the relevant data to keep the model remaining fresh, is also required. Once you have the data, you're going to use ML to

sidestep having to write the logic.
Here, Maps is anticipating information that you might want to know if you were in a multi-story building. What else can Maps anticipate?

Take the map on the right. Still in Japan, you glance at your phone in between meetings and notice that you're getting a recommendation. Maps is now connecting your past history that you like art, that you like museums, and that you're in Japan, to now recommend things to you. This is even more of a data problem. The ML is what allows the original limited how-to-get-from-A-B map to now become a virtual assistant.
Personalization of the Maps service is possible only with machine learning.
Machine learning is about scaling beyond hand-written rules. Then you start being able to do things that you could never achieve if you were writing hand-written rules.
Think back to your business. Your business analysts are essentially looking only at the bulk of your business. That's akin to the use case on the left. The stuff that everybody in the country needs. One set of rules for everyone. You might be thinking of ML as a way to do the things in the middle -- of being able to take the data you happen to have and training a ML model.
But think of ML as a way to get to kind of things on the right -- of being able to personalize your services for each of your customers. And notice the question at the bottom of the card on the right -- asking for user feedback to keep improving the model.
What's needed, though, in this transformation from the left (which is quite generic) to the right (which is quite personalized).

If ML is a rocket engine, data is the fuel

Data, and lots of it. The rules/models are actually quite simple. If ML is a rocket engine, data is the fuel. As you get into complex models and various ways of tuning a model to get better and better performance, it can be very easy to lose sight of a key point. Data wins. Every time.

https://pixabay.com/en/rocket-launch-smoke-rocket-take-off-67723/ (cc0)

Simple ML and More Data > Fancy ML and Small Data

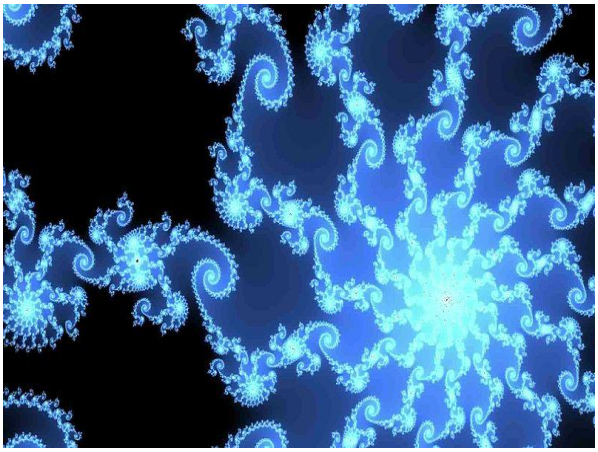


Image Source: https://en.wikipedia.org/wiki/Mandelbrot_set

Given the choice between more data and more complex models, spend your energy collecting more data.
That means, collecting not just more quantity, but also more variety.

For example, imagine that your data consists of these fractals. If you zoom way in, you won't see the patterns. You don't have enough data, so you'll end up fitting to very complex rules. As you get more and more data, hopefully you fill out the domain and the overall pattern starts to become much more evident.

An ML strategy is, first and foremost, a data strategy.

From: https://en.wikipedia.org/wiki/Mandelbrot_set

So, how do you get started on machine learning? Experience shows the typical customer journey, the one most likely to be successful, is to select a use case where you're doing manual data analysis today. This is what Global Fishing Watch did, a nonprofit that tries to identify poaching. They used to manually analyze fishing trips and they scaled up their processing using machine learning to the point that they could analyze 22 million data points daily.

There are several reasons why you want to go through manual data analysis to machine learning:

(1) If you're doing manual data analysis, you probably have the data already. Collecting data is often the longest and hardest part of an ML project, and the one most likely to fail. If you have the data already, your chances of success just went up.

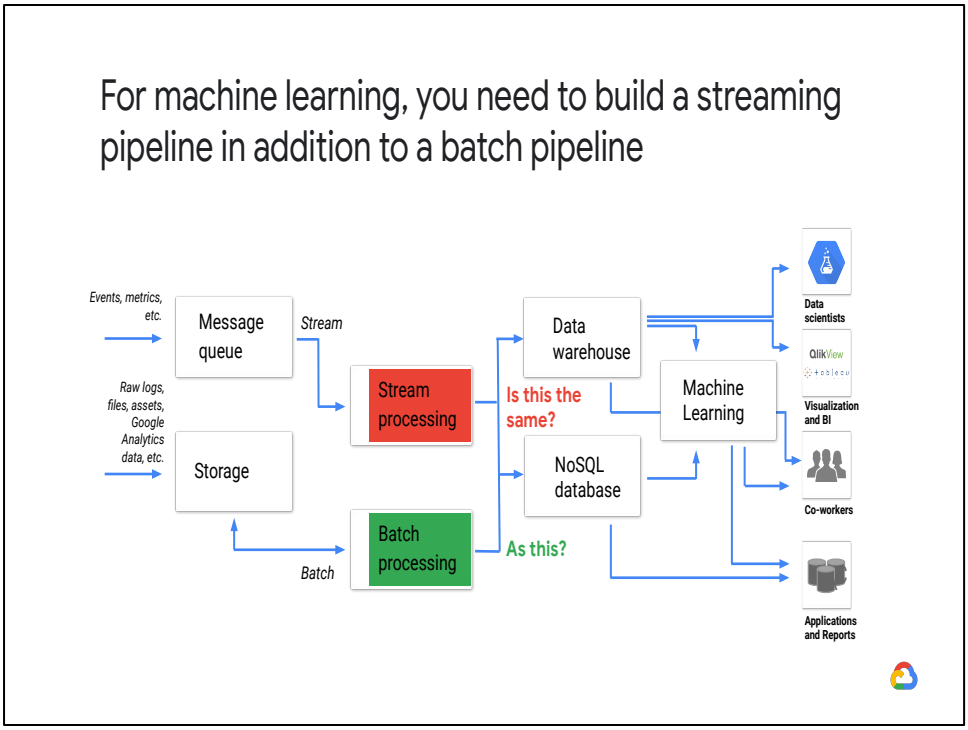
(2) Even if you don't have the data today, and your ML project involves first collecting and rating data, you want to go through a manual analysis stage. The reason is that if you can't analyze your data to get reasonable inputs towards making decisions, there's no point to doing ML. Manual analysis helps you fail fast and try new ideas in a more agile way. So, don't skip the analysis step.

(3) To build a good ML model, you have to know your data. Since that's the first step, go through the process of doing manual data analysis. Don't jump straight to ML. This is discussed more in the next module.

(4) Finally, ML is a journey towards automation and scale. You're automating manual analysis because you want it to scale. Perhaps, like Global Fishing

(1) Watch, you're manually analyzing a small fraction of fishing trips, and you want to automate this so that you can scale up to analyzing a great deal more fishing trips.

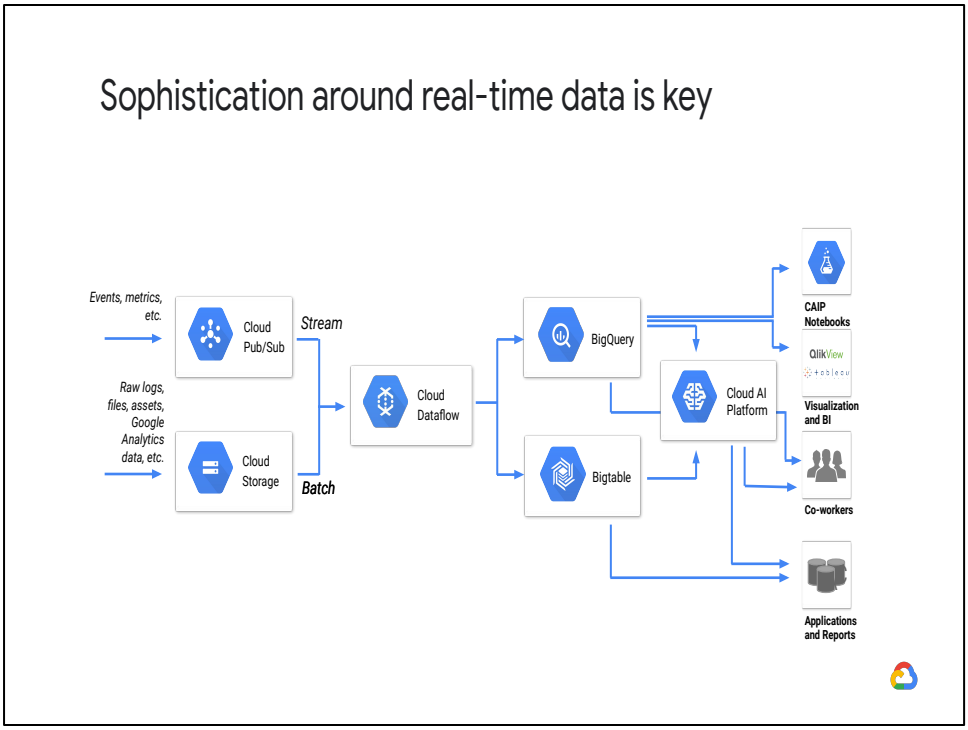
In short, if you can't do analytics, you can't do ML.



When you say ML to engineers, they keep thinking “training”. But the true utility of ML comes during predictions. That’s when you’re getting value from it. One key thing is that your models have to work on streaming data. You need to build up your streaming data sophistication. If you are thinking you could get away with doing things weekly, as batch processing, then guess what -- your business is only getting faster.

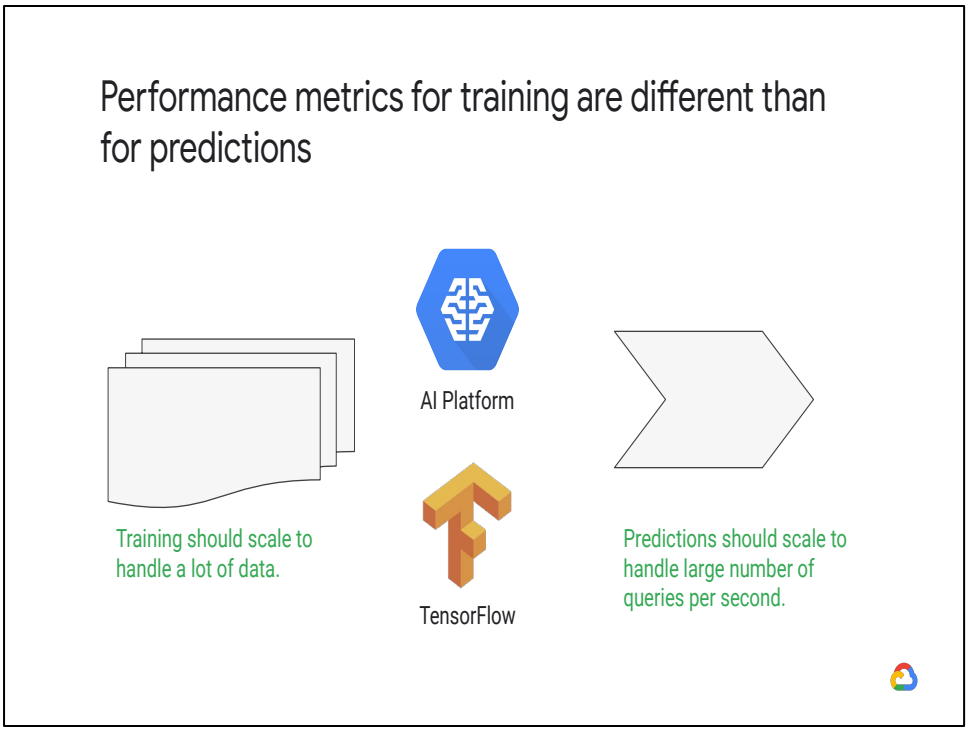
One common reason that ML projects fail is because of something called “training-serving skew”. This is where you had a certain system for processing historical data so that you could train it. Perhaps it was a batch processing system written by your data science team. Then, you have a different system that needs to use the ML model during prediction; the system that serves these predictions is probably written in something that your production engineering team writes and maintains. Perhaps it’s written in Java using web frameworks.

The problem is that unless the model sees the exact same data in serving as was used to do training, the model predictions are going to be off. That is the problem that is referred to as “training-serving skew”.



One way to reduce the chances of this is to take the same code that was used to process historical data (during training) and reuse it during predictions. But for that to happen, your data pipelines have to process both batch and stream.

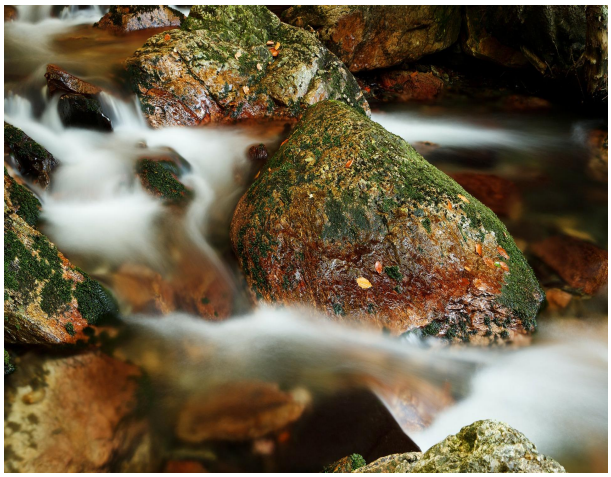
This is a key insight behind Cloud Dataflow, a way to author data pipelines in Python, Java, or even visually with Cloud Dataprep.
It's open-sourced as Apache Beam, where the B stands for batch and -eam stands for stream. A single system to do both batch and stream, because in machine learning, it's helpful to use the same system in both training and prediction.



The performance metrics you care about change between training and predictions as well. During training, the key performance aspect you care about is scaling to a lot of data. Distributed training, if you will.

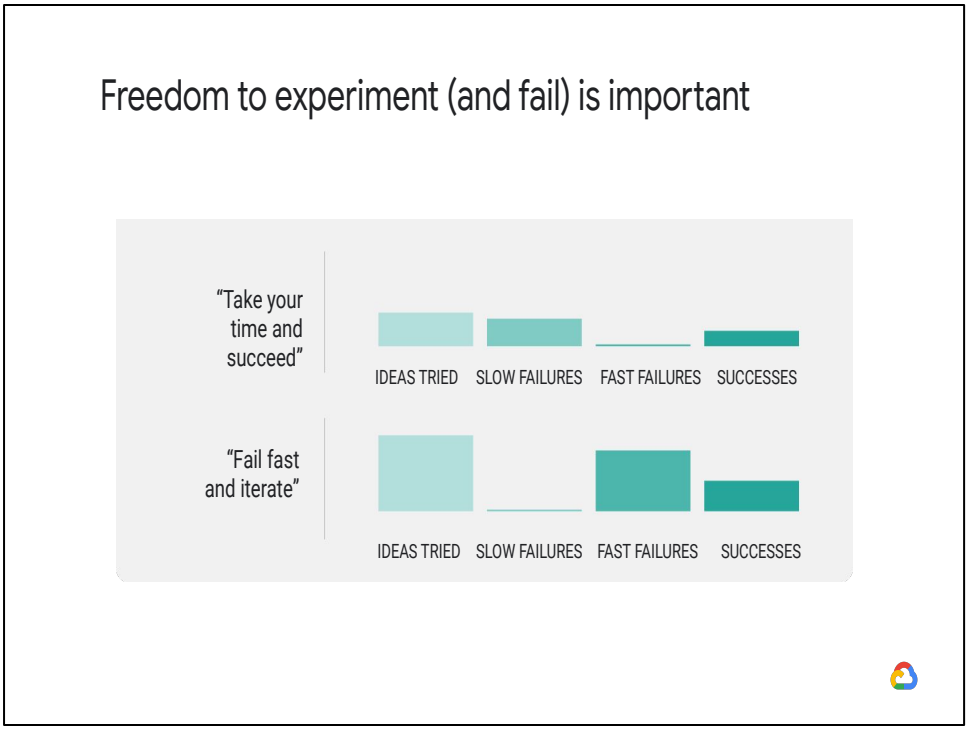
During prediction though, the key performance aspect is speed of response (high QPS). This is a key insight behind TensorFlow. Lots of ML frameworks exist for training, but not so many are equally capable of operationalization.

Connect simple ML models into a pipeline



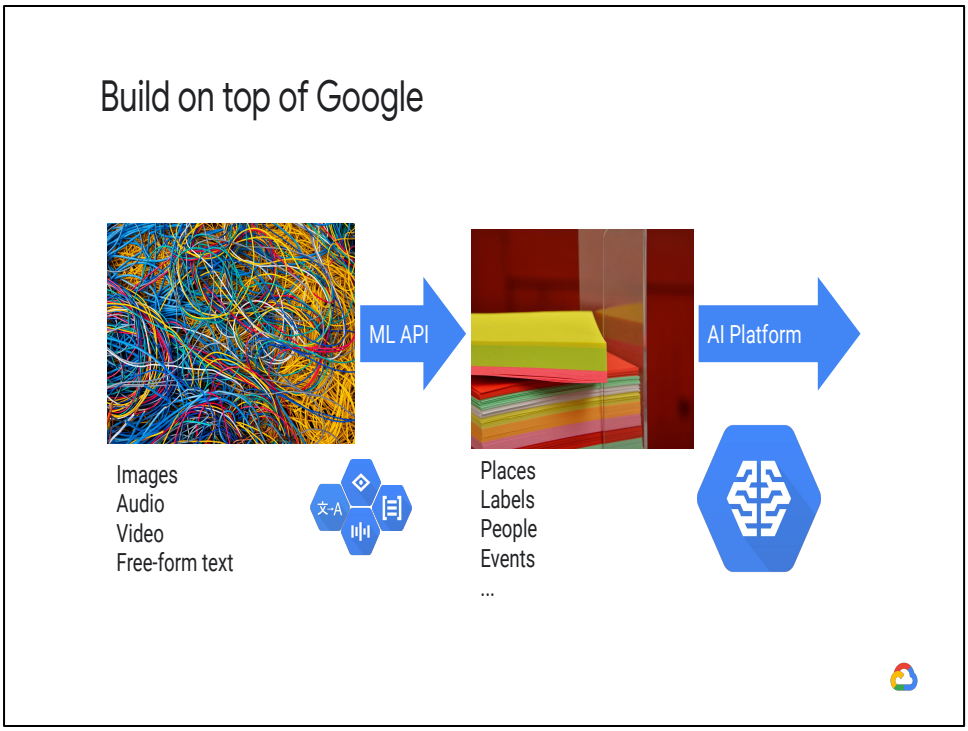
If there's one thing you should take away from this module, it's that the magic of ML comes with quantity, not complexity -- many small ML models each of which is simple. Think of the Maps example.

https://pixabay.com/en/blur-blurred-cascade-fall-flow-21653/ (cc0)



If you're building many ML models, and planning for many more that you may never build, you want to have an environment where you fail fast.

The idea is that if you are failing fast, you get the ability to iterate. This ability to experiment is critical in the realm of machine learning.



When you say data, most people immediately think “structured data”, in their database. But 90% of enterprise data is unstructured. Think emails, video footage, texts, reports, catalogs, fashion shows, events, news, etc.

Dealing with structured data has gotten a lot easier because of the pretrained models talked about. Think of an ML pipeline as a way to deal with unstructured data. Take unstructured data, pass them through ML APIs, and then you’re left with entities, places, labels, people, things that you can build a simpler ML model out of.

https://pixabay.com/en/list-zettelbox-note-leaves-stack-1925395/ (cc0)

Agenda

What is ML?

What kinds of problems can it solve?

Infuse your apps with ML

Build a data strategy around ML

Use ML creatively, to delight your users

Fulfill user intent, using ML creatively, to delight your users. The ability of ML to scale, to be personalized, gives you lots of opportunities to do this.

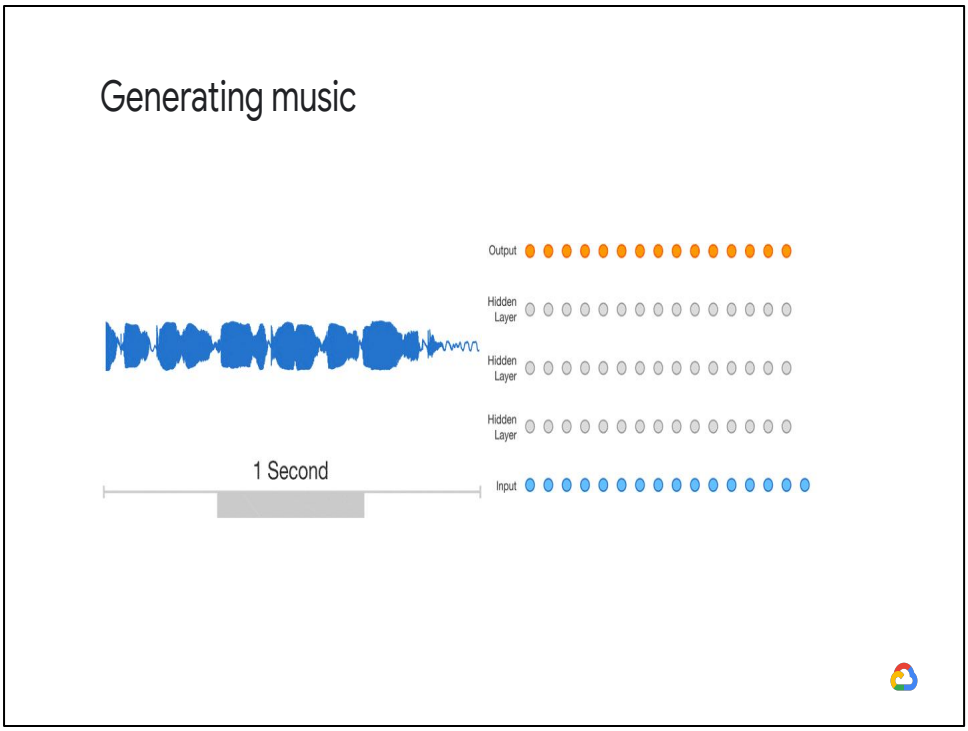
What's the customer experiencing right now?

For ML to be magic, you should keep in mind the word "delight". Delight your users. Anticipate their next need.

Did the card get canceled?, Did the flight get delayed?, etc. Or, did they just walk into your store and see an empty shelf?

Does your system automatically find the "right" action to take for this customer? Rebook them? Offer them a meal voucher? Suggest that you're happy they are wearing your brand of sneakers?

https://pixabay.com/en/supermarket-shopping-empty-shelves-665049/ (cc0)



Fulfill user intent in interesting ways using the capabilities of ML.

Users want background music in their videos, but using popular songs is a copyright violation. What if you generate music according to their specs??

Classic piano music generated by WaveNet:
https://deepmind.com/blog/wavenet-generative-model-raw-audio/
Virtual girl images generated by GAN on TF: https://goo.gl/xkzbYI
For example, game companies are trying GAN for generating avatar images for online game players.

Your business can benefit from ML

Infuse your apps with ML

Simplify user input.
Adapt to user.

Fine-tune your business

Streamline your business processes.

Create a new business.

Delight your users

Anticipate users' needs.
Creatively fulfill intent.

Image (Infuse ML): https://cloud.google.com/products/machine-learning/
Image (Fine-tune business) cc0:
https://pixabay.com/en/electrician-electric-electricity-1080554/
Image (Delight Users) cc0:
https://pixabay.com/en/youth-active-jump-happy-sunrise-570881/

Lab

Non-traditional ML use case

Image (Thinker): https://pixabay.com/en/art-auguste-rodin-bronze-famous-1301872/

Lab Steps

1. Think back to an existing application at your company.
2. Brainstorm on how you could replace parts of it by ML (think beyond user interface elements).
3. What are some of the benefits to doing so?
4. What kinds of data would you collect if you wanted to do this?
5. Are you collecting that data today? If no, why not?

Image (Thinker): https://pixabay.com/en/art-auguste-rodin-bronze-famous-1301872/

Welcome to How Google does ML.

Machine Learning with TensorFlow on GCP

How Google does Machine Learning

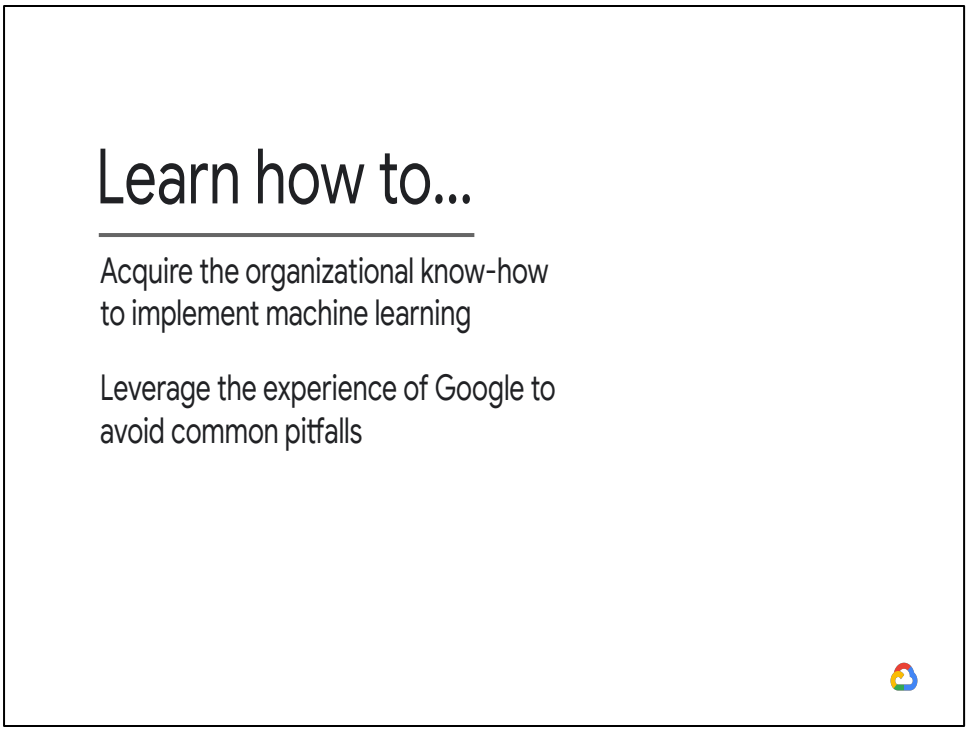
Launching into ML

Introduction to TensorFlow

Feature Engineering

The Art and Science of ML

This is the second module in the first chapter of the *Machine Learning with TensorFlow on GCP* specialization.



In this module, you'll learn how to acquire the organizational know-how to implement machine learning, as well as leverage the experience of Google to avoid common pitfalls.

You will be introduced to the organizational know-how acquired over the years, as Google has built probably more value-generating ML systems than any other company. There's no math, no TensorFlow, and no Cloud. You're going to put your business hat on, and look at ML the way the owner of a business, someone responsible for setting its direction, would.

Agenda

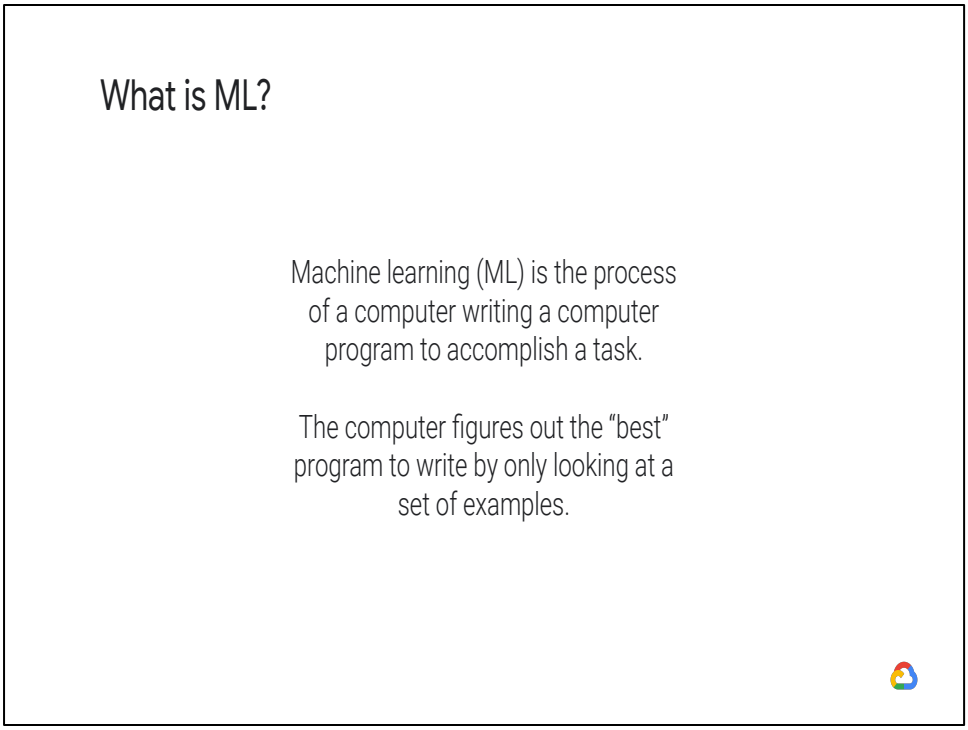
The ML surprise

The secret sauce

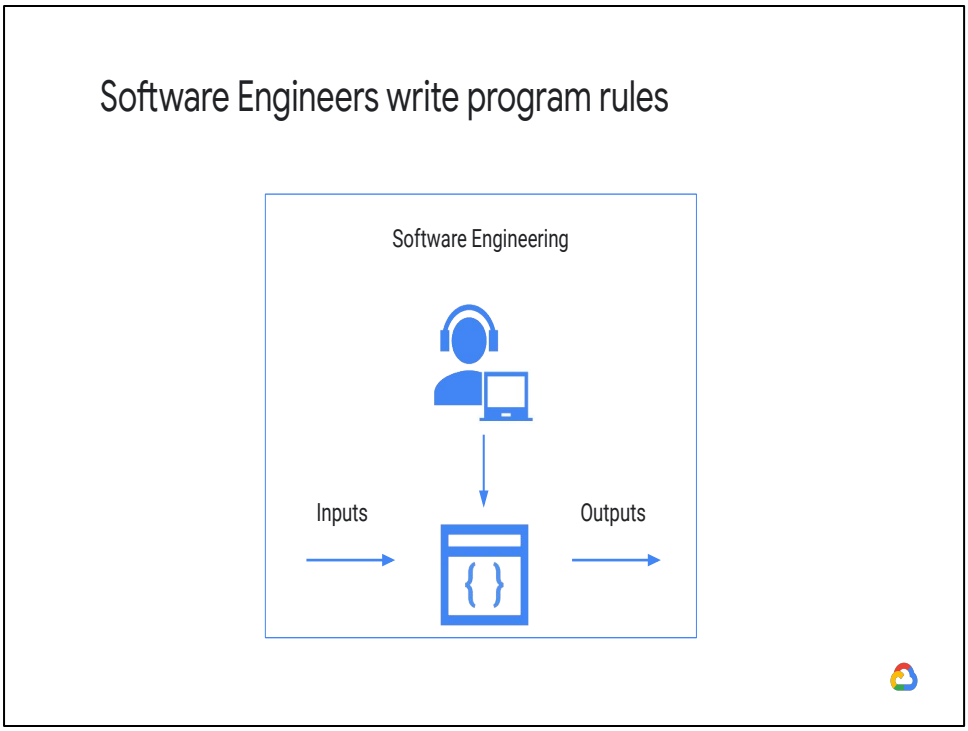
The ML and business processes

The path to ML

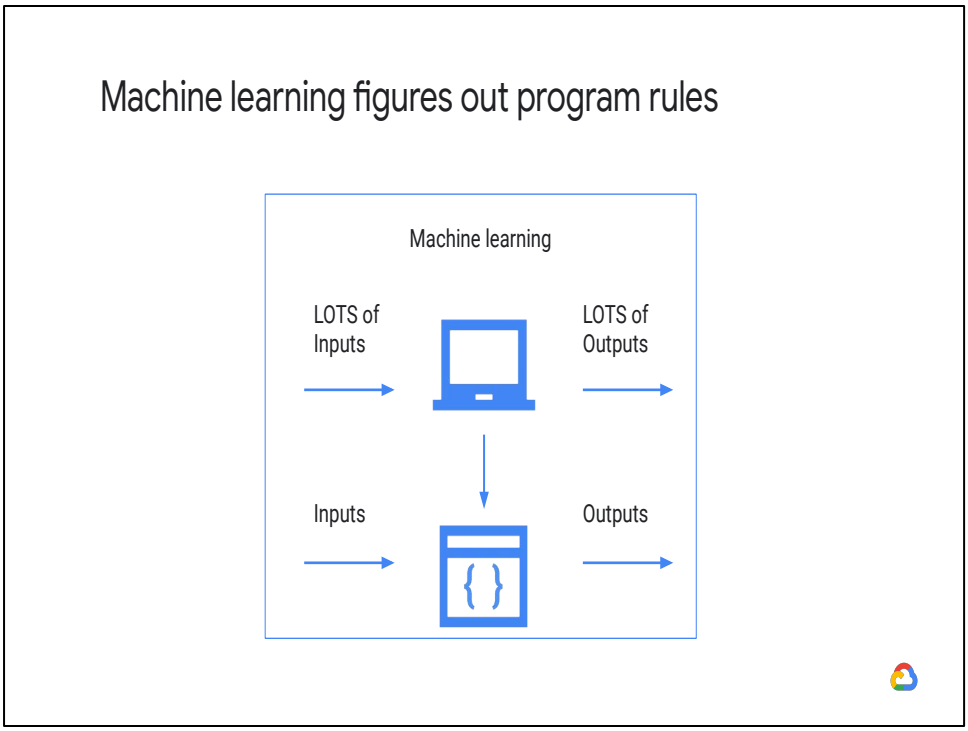
End of phases deep dive



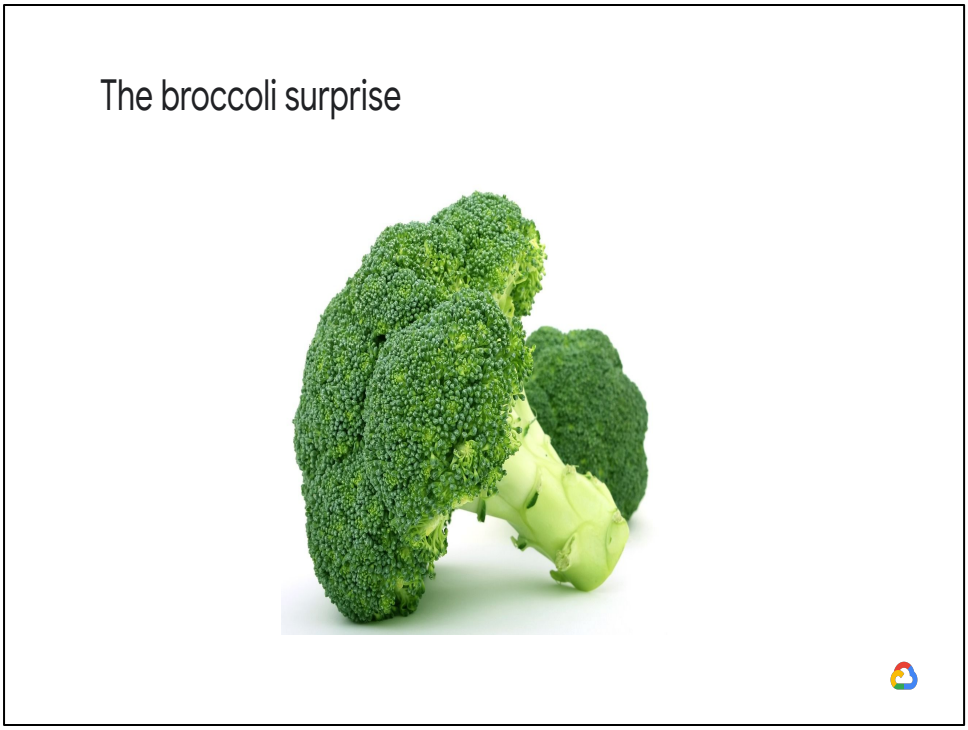
By ML, or machine learning, what I mean is this process by which one computer writes a computer program to accomplish a task. The computer that is doing the writing figures out with the best program is by only looking at a set of examples.



So, let's compare this to normal software engineering. Kind of this more typical approach, we have a human who analyzes the problem, writes a bunch of code, and then, this code becomes a program that can translate inputs to outputs. Maybe it's a calculator and it knows how to take two numbers and add them together. So, three and four produces seven.



What happens in machine learning? Machine learning, we're kind of going to pull out the software engineer. And instead, we're going to use another computer that is only going to see many, many examples, many inputs paired with the desired output. And from these, that computer will figure out with the best "program" is to write. Now, obviously, this is not a technically correct fully mathematical academic definition of ML. That's fine. This is just going to give us the framework we need to have a conversation about ML in businesses for today's course.



This is actually a true story from my undergrad days.

So, this new ice cream shop opens up, and a couple friends and I decide to walk over. We get in there and are reading off the flavors: peach, mint, chocolate and a *broccoli surprise*

Obviously I gotta try that one, so I get a sample. Its white with these little green spots in it. Its creamy, rich and delicious. Quite the surprise for anything with broccoli in the name. So I ask lady behind the count “what’s the surprise?”

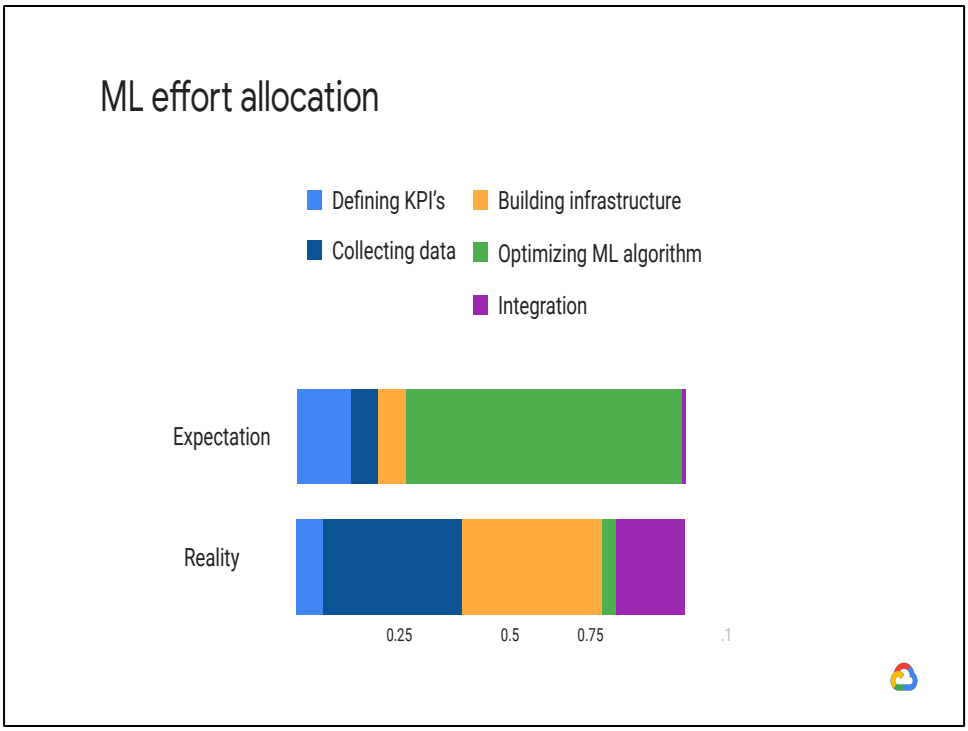
Image cc0 (Broccoli)
https://pixabay.com/en/appetite-broccoli-brocoli-broccolli-1238250/

The broccoli surprise

She just smiles at me and says “there’s no broccoli.”

Notes: The intent here is to cue up the idea of surprise as mistaken expectation, which transitions into the common misconception about ML effort distribution

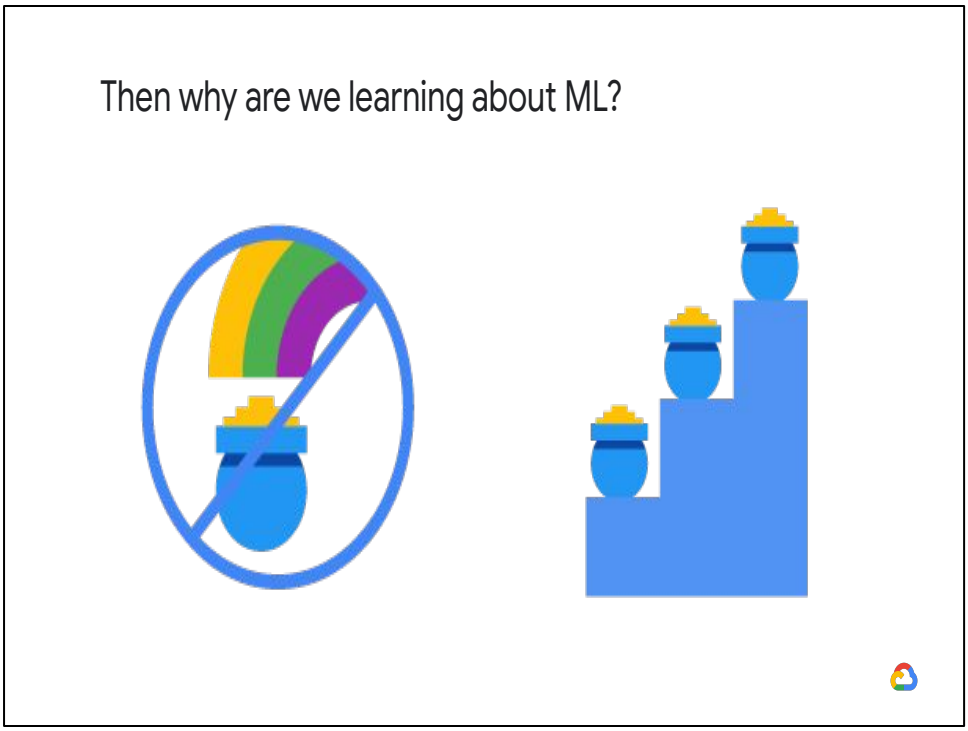
Image cc0 (Ice Cream)
https://pixabay.com/en/ice-cream-ice-cream-parlor-dessert-410330/



So, what I'm going to do is show you a couple of bar charts that portray how you would spend your effort in a variety of different tasks as you build a fully end to end ML system in your organization. We have things like defining the KPI, what you should even be trying to accomplish, collecting the data, building the infrastructure, optimizing the ML algorithm itself, and then integrating with the rest of the preexisting systems at your organization.

Now, very informally, but from many conversations I have with new ML practitioners internally and with our external partners, I find most people really tend to focus just on optimizing the ML algorithm. They want to make sure they have the newest, coolest thing right out of the papers. They've tuned all the right hyperparameters. They have the right number of convolutional layers. A lot of very technical details about the ML. But when I look and I talk to practitioners inside Google that have had great success building these big systems, I find a very different story. In fact, what I find is that how optimize the ML algorithm takes a much smaller segment of effort than people expect. I've never found anyone who overestimated how hard it was going to be to get that data collection right in the first place. And we should really pay close attention to that data collection. And what I would say is infrastructure building, making sure that we can train our model many, many times and automatically and smoothly or making sure we can serve that model at scale to our end users. And in fact, these kind of more core, almost software tasks, end up really dominating how people spend their time and effort when they build these successful ML systems. And the final point is, once we get to ML, we have another advantage that everything about our users or operations are so well measured that we can actually spend a little bit less time

defining KPIs, maybe a little less organizational effort. Because it's no longer a theoretical approach. We're no longer relying on someone's intuition from a previous slide for some market research. We just measured everything we need to know about our users and this gives us great insights to figure out not what intermediate KPIs to use, which is how to get to the right, ultimate one, like customer lifetime value or net present value.



So, why are we learning about ML if the surprise is ML is not so important?

ML is great because the path we take to get to it is going to yield a lot of value all along the path. Maybe not every problem ends in ML but many will. And even those that don't will benefit from going down this journey.

In this lecture, we're going to talk about The Secret Sauce. So Google is going to share The Secret Sauce with you.

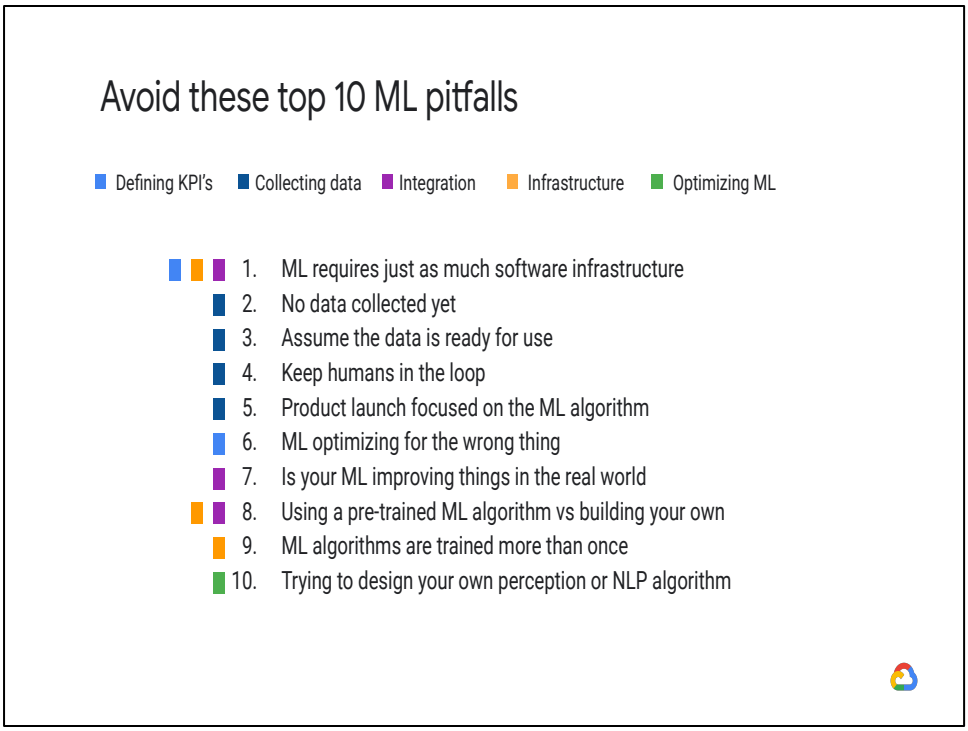
But that secret source is not code, it's not just an algorithm, it's actually this organizational know how that we've acquired over years of managing probably more value generating ML systems than any other company in the world. So if we're going to share this organizational know how, why start with technical ML skills? Well, we want you to become great ML strategists.

Image is Google Brain Logo cc0 taken from:
(https://cdn.slidesharecdn.com/ss_thumbnails/k2jeffdean-160609173832-thumbnail-4.jpg?cb=1465493958)
Image cc0 (Secret sauce):
https://pixabay.com/en/kagyana-strapatsada-gdarta-2955466/

Get your hands dirty by practicing with technical skills

And to do that, we believe that you need to get your hands dirty. You actually need to go up and you need to build some of these systems and learn about them. And the good news about that is that these technical ML skills that you're here looking for on course era, well, they're mostly software and data handling skills anyway. There are things you may already be very comfortable with. And as we talk about these technical skills, it also gives us an opportunity to leverage Google's experience to help you avoid some of these common pitfalls.

Image cc0 (dirty hands):
https://pixabay.com/en/soil-dirty-hand-people-man-guy-2564678/



What are some of these common pitfalls? I'm glad you asked. So here is our kind of click baity fun, top ten pitfalls organizations hit when they first try ML. And here's a list, very informally I've aggregated after several years of talking with new ML practitioners that come to us and they say, "We're so excited to this great new thing, it's going to be awesome." And then they might fall into some common pitfalls. I've seen it at Google, and I've seen it with our partners as well. First one, perhaps one of the most common, you thought training your own ML algorithm would be faster than writing the software. Usually, this is not the case. And the reason is that to make a great ML system beyond just the algorithm, you're going to need lots of things around the algorithm like a whole software stack to serve, to make sure that it's robust and it's scalable and has great up-time. And all of this, you're going to have to do for software anyway. But then if you try to use an algorithm, you put in additional complexities around data collection, training, all of that just gets little bit more complicated. So usually, we really push people to start with something simpler in software only.

Next one, one of my favorites. You want to do ML, but you haven't collected the data yet. Full stop, you need the data. There's really no use talking about doing great ML if you have not collected great data or you do not have access to great data.

And let's say you do have that data, you've been logging in for years, so it's written on some system that someone in another department controls, but you haven't looked at it, willing to bet that if you haven't looked, that data is not really ready to use, and it goes even beyond that. If there's not someone in your organization who's regularly reviewing that data or generating reports or new insights, if that data is not generating

value already, likely, it's not the effort to maintain it is not being put in and data has this kind of magical way of going stale. Of all the clients I've ever talked to, I've never met one who overestimated the amount of effort it would take to collecting clean data. No one has ever said that was easier than I expected, expect there to be a lot of pain and friction here.

What's the next one? You forgot to put and keep humans in the loop. So when we get into these ML systems that start to perform core tasks or core business processes in our organizations, they become really important. And appropriately, organizations become risk averse around these systems because they are the breadwinners of the organization and then becomes very important to mitigate this risk. And one of the myriad of ways we do that is we keep humans inside the loop so that they are reviewing the data, handling cases the ML did not handle very well and curating its training inputs. And we're going to talk about this more later, but this is a feature of every production ML system I know in Google, is that it has humans in the loop.

What about this one? You launched a product whose initial value prop was its ML algorithm instead of some other feature. So this is a problem because A, your users probably don't care if what you're giving them is the ML, they just care if it's got that new cool feature or if its recommendations are really good. And, if you launch something whose initial value prop is just ML, it has new data to operate on. It needs lots of users to generate that data so it may learn how to interact better.

What about you made a great end ML system, it just happens to optimize for the wrong thing. So imagine if Google search was optimizing for, let's say a user engagement as measured by how often someone clicked on search results. It sounds good. We want our users to like our product, we want our users to stay engaged. But if we optimize for how often they click, maybe then the ML algorithm will learn to kind of serve bad content because it forces users to come back, keep clicking. So we always want to be careful about optimizing for something that's pretty good, need not be perfect, but we will always want to look out for perverse incentives.

So what happens if you forget to measure if your ML algorithm is actually improving things in the real world? You put it out there, you turned it on, it serves users, but you can't tell how much better it is, you can't tell if there's any uplifting customer engagement, or lifetime value. That's always really worry some because then how are you going to go back to your boss or your boss's boss and say, "Hey, I want to do this for another product if you cannot show the impact of the success."

And then I've seen a couple of customers to this next ones, you confuse the ease of use and the value add of somebody else's pre-trained ML algorithm with building your own. So Google Cloud has a couple what we call ML APIs. For instance, with vision, you can send it an image and it will perform image classification on some predefined labels. Well that's great, it's super easy to use. You don't have to worry about any

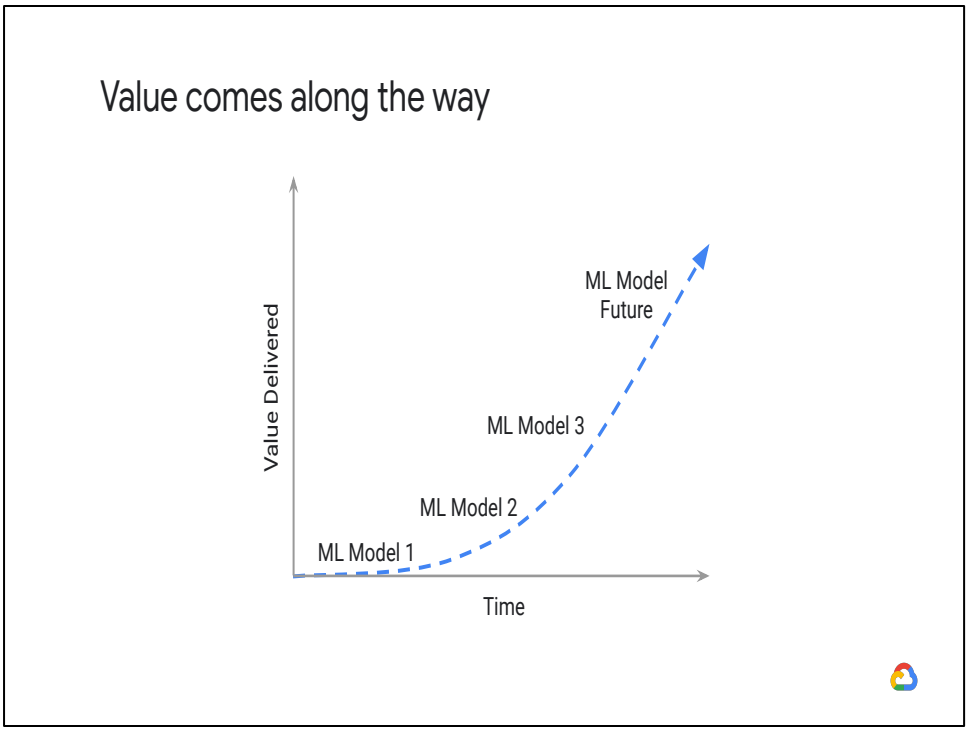
infrastructure, or any training data, or any data collection, very easy to use. It is a very different ballgame than if you went to start to build your own, especially if you want to do your own ML algorithm that does not kind of come pre canned, it's a lot more effort.

We thought after we research that production ML algorithms were trained only once. You're like, "Hey, it's on my laptop, it's doing great on that data set. I'm basically done." No, you're probably about 10 percent of the way through. It turns out that if you're going to have an ML algorithm that's going to be part of your core business processes, it's going to be retrained many, many times and you're going to want to invest the effort to make that process very easy and seamless.

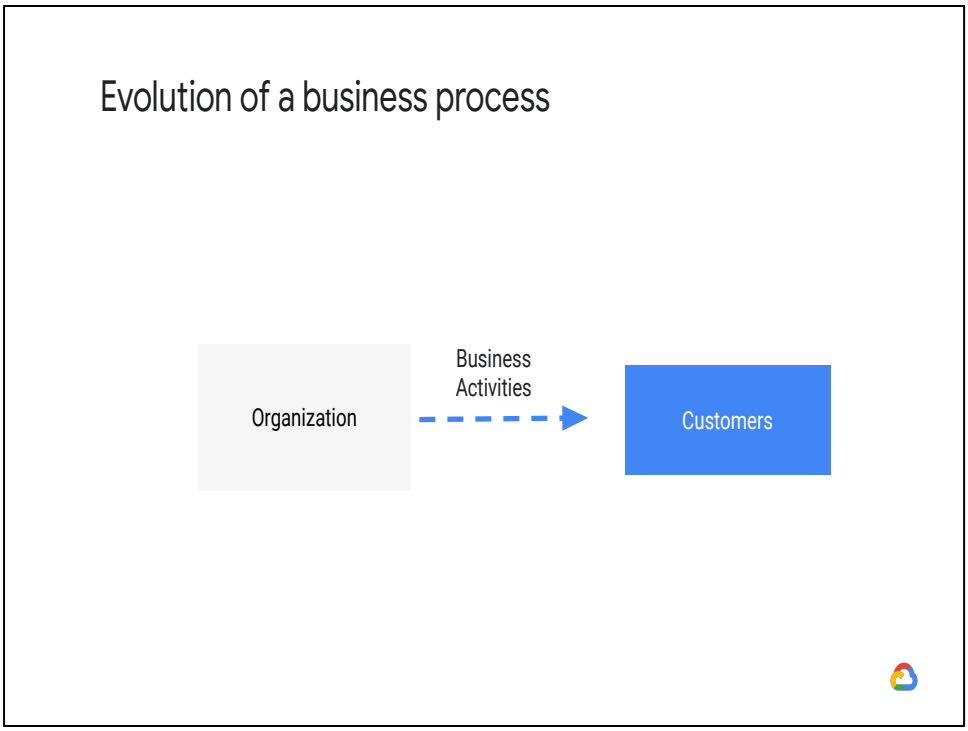
And the final one is actually the only one these I have that addresses a confusion about the challenge involved an opera optimizing the ML algorithm, and that's, you want to design your own in-house perception, i.e. image or speech, or NLP classification, or that's natural language processing. So these are kind of a peculiar pitfall in the sense that they seem they're much easier than they really are. And in fact, all the algorithms we have to address these are very highly tuned from decades of academic research and you should almost always take one off the shelf, already made or already kind of defined, instead of trying to do your own research, it's very expensive.

So that's a lot about it. That's a lot of pitfalls. That's a lot of problems. What's the good news? So the good news is, most of the value comes along the way. As you march towards ML, you may not get there, and you will still greatly improve everything you're working on. And if you do get there, ML improves almost everything it touches once you're ready. And think about this, if the process to build and use ML is hard for your company, it's likely hard for the other members of your industry, right? And once you have that ML enable product or internal process, it's going to provide the users or the consumers of that process great experiences that become very hard to duplicate or catch up to because of this beautiful feedback loop where it's collecting more data and learning all the time. So, I would like to double click into this idea that value comes along the way. I know it's tempting to try to jump to a fully machine learned, automated end to end, auto magic everything solution. We all want to make this leap, but it usually doesn't lead to great products organizational outcomes. I've seen that in Google, and I've seen that in our partner organizations as well. So what I want to do now is review a more realistic path and all the great things that come along the way.

Image cc0 (Journey):
https://pixabay.com/en/mountain-road-winding-road-travel-1556177/
Image cc0 (Gold): https://pixabay.com/en/gold-is-money-gold-bar-shop-gold-2430051/
Image cc0 (Difficult):
https://pixabay.com/en/magic-cube-mess-hand-puzzle-toys-2399883/
Image cc0 (Jumping):
https://pixabay.com/en/youth-active-jump-happy-sunrise-570881/

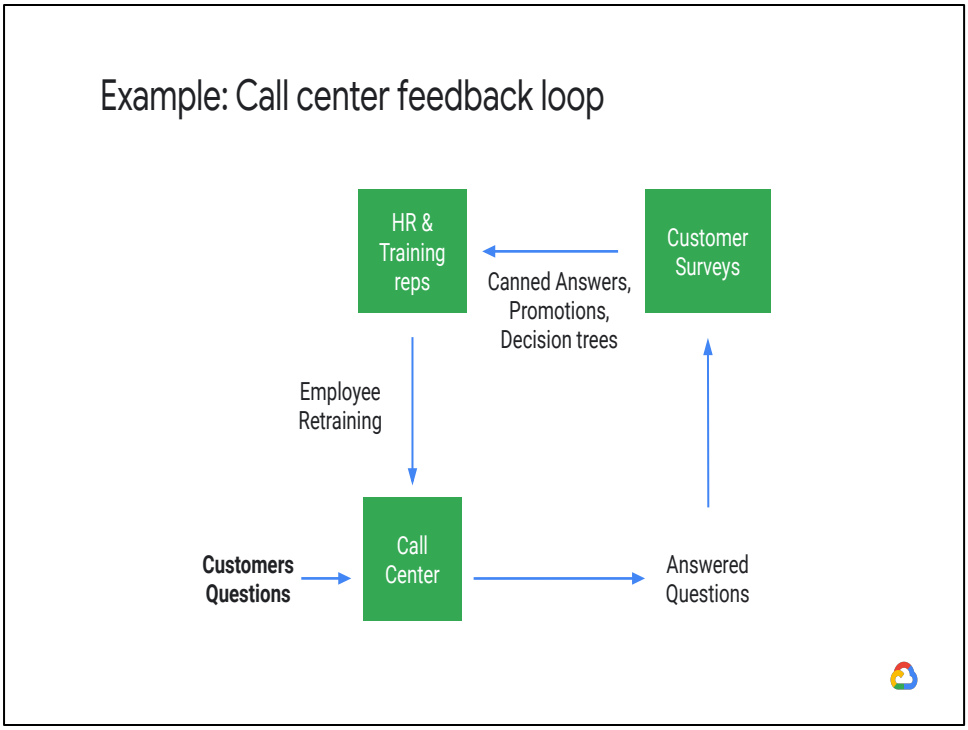


So what I want to do now is review a more realistic path and all the great things that come along the way.

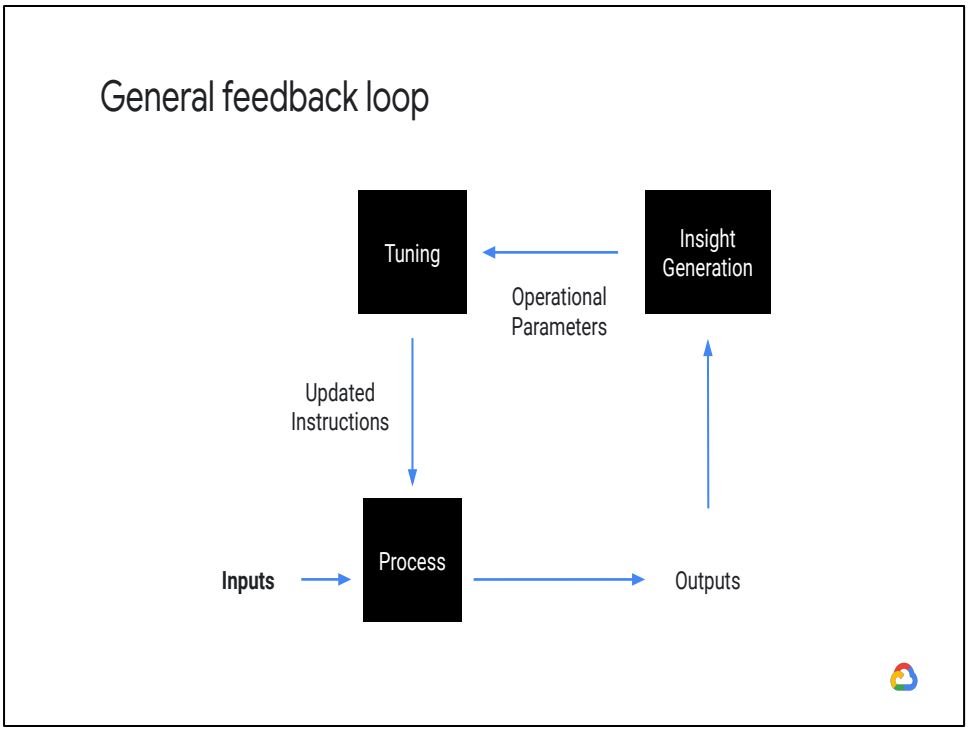


So, when we think about how we're going to get from a No ML to ML solution in our organization this path, I really want us to think about the evolution of a business process. And here when I say a business process, I'm talking about any set of activities a company must do directly or indirectly to serve customers.

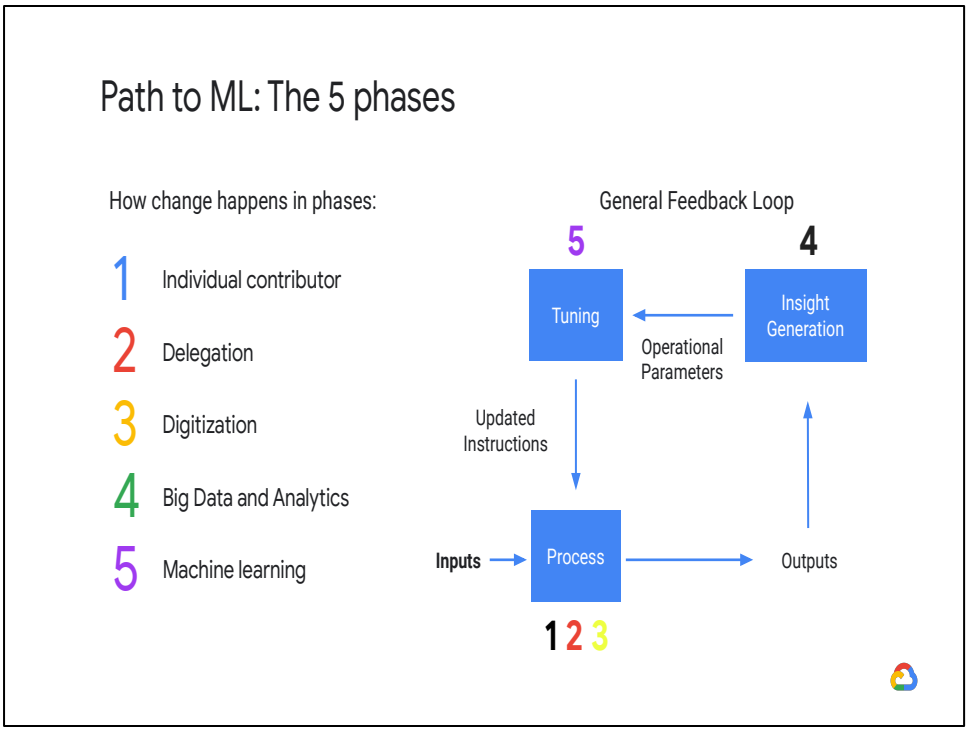
Organizations must continually improve these processes which they do through a feedback loop, and this is really critical.



Oftentimes, when we think of business processes, we forget that almost every one of them has a feedback loop, and understanding that loop becomes really important to understanding the role ML plays inside a large organization. So, let me give you a concrete example. Let's say we have a call center and the call center takes customer questions, and they produces answered questions so you call your favorite local telecom, and you say, "There's a problem with my Wi-Fi router." And the call center answers it for you. But the interaction doesn't end there. We all know that after that call center hangs up they say, would you like to answer the survey or they email you. And what they do if you actually answer the surveys, is they take all those answers, and they extract new insights from them. They get new canned answers, new product promotions, or decision trees on how to handle future calls. And then those new insights, they get fed into HR or training reference manuals, and the employees are retrained, and those retrained employees may now go answer new customer questions. And this is this kind of feedback loop that allows us to convert operational expertise into better future outcomes.



Now here, I'm showing you a more general view of this. So here, we have some input and some output to a process, this looks similar to our example of what ML is compared to software, that's not an accident. And now, what we do is we take this output, and we regenerate new insights from it, and these insights are going to give us some new operational parameters such as a new canned answers or a new product promotion. And then we're going to tune the original process with updated instructions. And here again, we have this flow but in a much more general way, they could be applied to almost any business process in any organization.

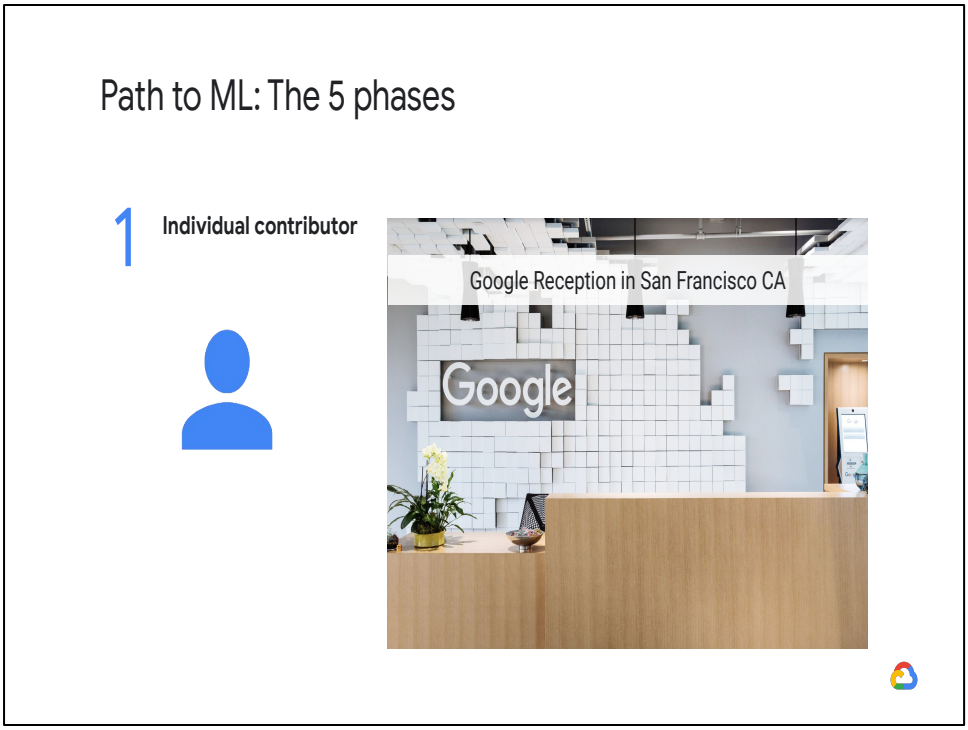


And so, when we think about the path to ML, I want you to think about how we're going to automate each one of these boxes, each one of these cornerstones and the business process tuning. So, in the first step for anything, I'm going to explain more of these steps later slide.

But, the first step of any new business processes just individual contributor, one person doing it, and then you get multiple people doing it, and then you digitize that process, and these steps one, two, and three, they all affect the core process itself.

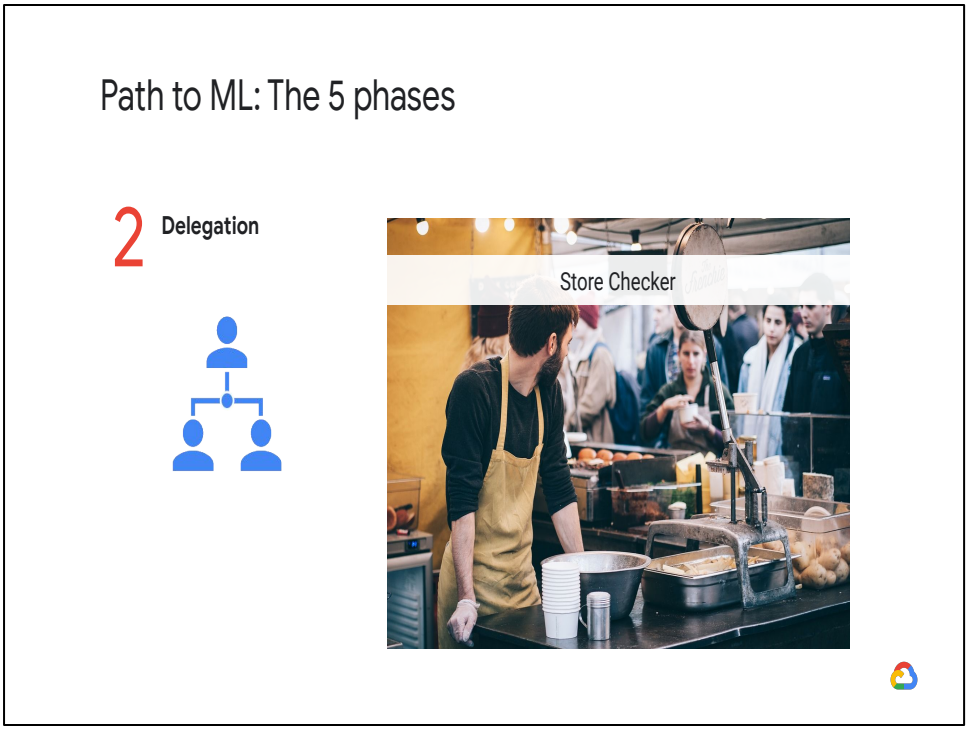
But, as you know in the last decade or so, Big Data, and Analytics, and Machine Learning have become very popular and very impactful. And what happens there is we're trying to automate the inside generation phases and the tuning phases, and thus, we have automated the entire feedback loop.

So business processes that eventually end in ML, typically go through five phases. Now, they don't have to spend the same amount of time and each of these phases, but skipping these phases as we'll see later, usually isn't a good idea.



, the first one, what do I mean when I say, "individual contributor"? So, a task or business process that's in the individual contributor phase is performed by a single person, and a great example is like the receptionist inside the office building. This person, call, answer the phone, maybe points people towards the bathroom, just one of them, and the task is not parallelized or scaled at all, usually it's very informal.

Image from Chris McQueens photo shoot in SF Office
https://drive.google.com/corp/drive/u/0/folders/0B0JE7OIPZUWDWVd5NkMtTzJVMjQ



And then what happens is as the task or the business process becomes more important to the company. Usually, we start to delegate, we get multiple people who are all performing the same task in parallel. A good example would be like a store checker. And what happens when we start to delegate is we have to start to formalize the role and put in rules, so that each store checker starts to behave a little bit more like the others. So, there's some repeatability in the task.

Image cc0 (store checker):
https://pixabay.com/en/people-men-women-crowd-cook-bread-2595757/

Path to ML: The 5 phases

3 Digitization

ATM

Then we get to digitization. A little bit of a marketing buzzword. But, what I mean is that we take the recore repeatable part of a task or a business process, and we automate it with computers. Great example is an ATM. ATM's can't do everything. Right? You can't open a mortgage through an ATM, but you can withdraw a cash. And because that cash withdrawal part of that business process, where the interaction with the user is so repeatable and so well automated, customers get a very high quality of service using ATM's, and how many of us actually would walk into a bank to extract $40? Almost no one. But after we digitize, what happens next?

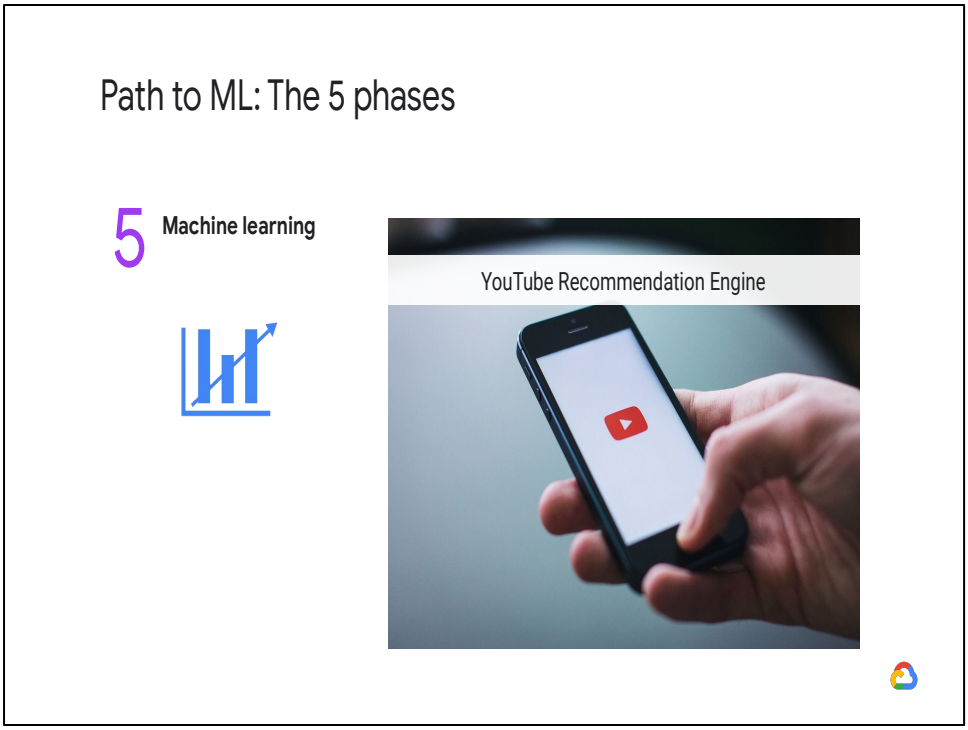
Image cc0 ATM (https://pixabay.com/en/atm-money-cash-payment-finance-2923515/)

Path to ML: The 5 phases

4 Big Data and Analytics

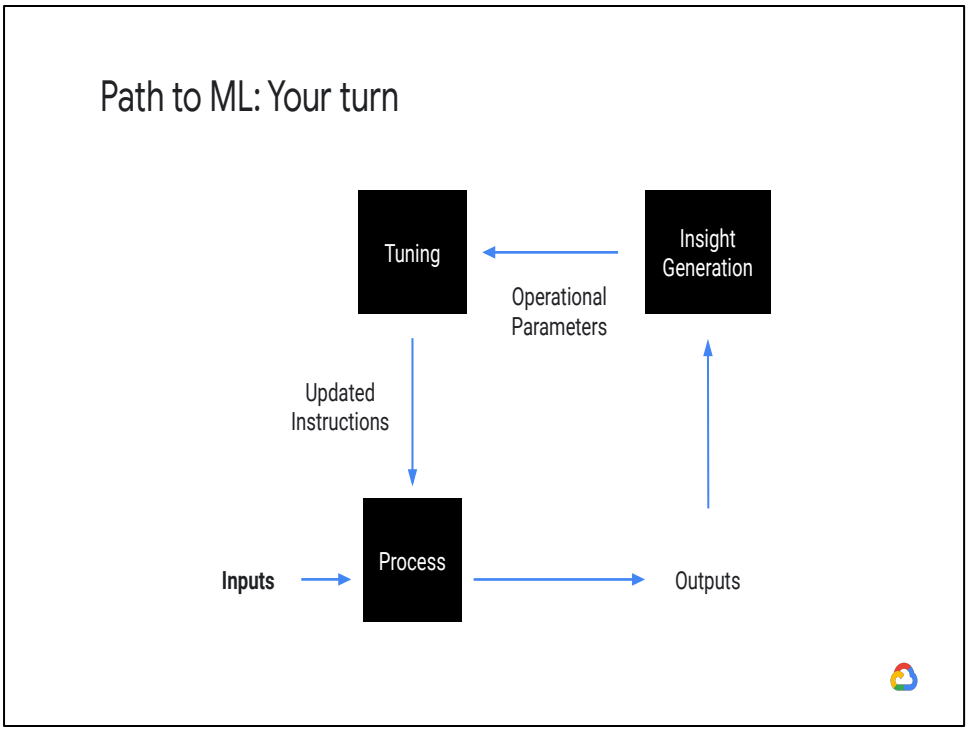
Now, we move into big data and analytics, and the idea is here we're going to use a lot of data to build operational and user insights. So maybe, when I say operational, a good example will be like, Toyota manufacturing. So, Toyota is famous for their lean manufacturing philosophy, where they kind of measure everything about their construction or their facilities, and then they use that to tune each little knob in the process to get better and better outcomes, and faster and faster cars, faster and faster time to delivery. And you could do this for your internal operations or you could do this to learn about your external users, and this would be like marketing research on steroids.

Image cc0 (car): https://pixabay.com/en/car-engine-motor-wires-suspension-2773263/



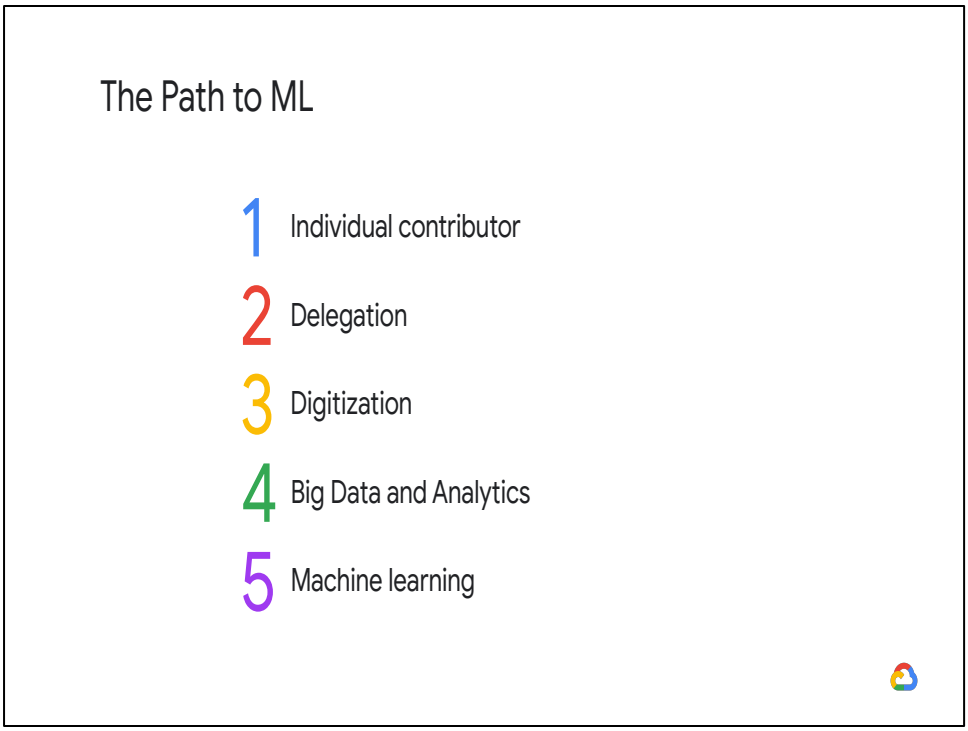
And then, of course, we get to machine learning, which is kind of we're going to represent the last phase in the path the ML. And here, we're going to do is going to use all this data that we had from the previous step. We're going to automatically start to improve these computer processes. And a big example here is YouTube Recommendations. As you click through YouTube, and you watch different videos, and you like them, or you don't like them, or you watch to the end or not, the algorithm is learning in the background. What are good videos, what kind of videos you like, how you are different or similar to other users.

Image cc0 (store checker):
https://pixabay.com/en/people-men-women-crowd-cook-bread-2595757/

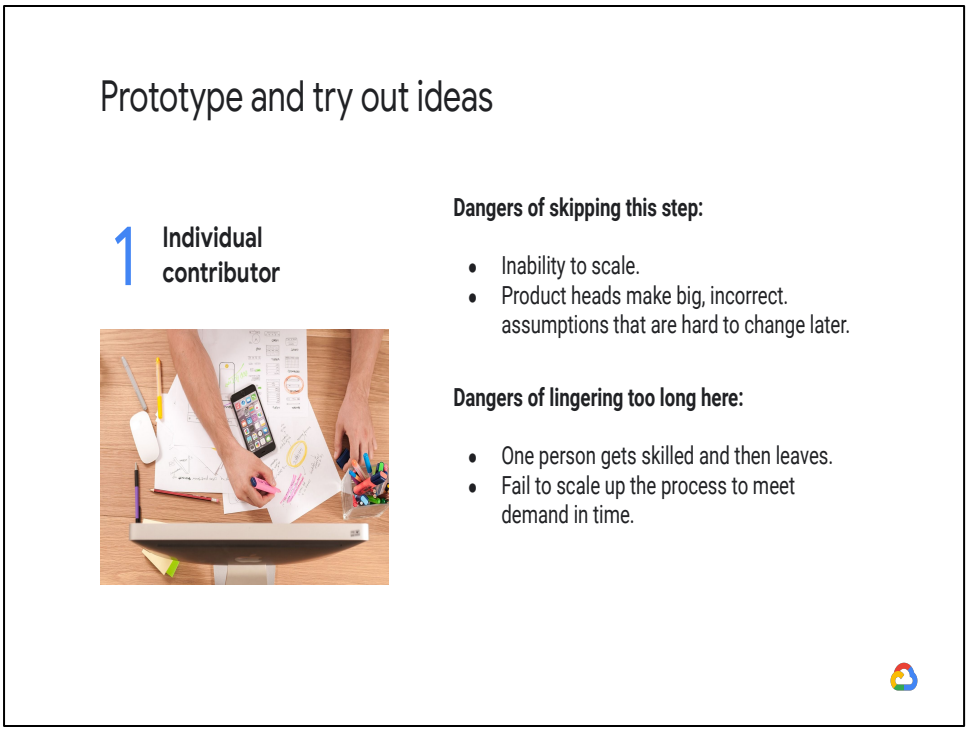


What I want to do is we think about this path MLs. I want you to take a moment, I want you to sketch this diagram for a specific example from your organization. It doesn't have to be an ML example. Maybe you have digitized part of this business process, but not all of it. What phases of the pathed ML is your example in? Do you have another example too that's in a different phase?

Image cc0 (store checker):
https://pixabay.com/en/people-men-women-crowd-cook-bread-2595757/



Let's examine each step a little bit more closely. Let's clarify the road map that a process or product goes through as it gets further automated and approaches and becomes ML. Let's keep in mind as we go through this, some examples from your company or your organization. Have they gone through any of these steps? How did it work out? Where did the company spend most of its time or where did it skip entirely? And do these phases seem hopeful like a constructive framework to think about? Or do they seem more like obstacles?



For the first one, let's start with the individual contributor phase and let's examine it. So this is a great opportunity to prototype. Almost every product or process starts here. It's cheap, it's flexible, great for trying out many ideas, failing quickly and learning from that.

There are some dangers of skipping this step. There's the inability to get organizational investment to scale up to a bigger effort. How are you going to convince your boss that you need a hundred people or you need to build this huge software system, if no one's even tried it, kind of manually? No one's then even that most basic prototyping. Perhaps even more worrisome, is that your organization does jump right on and then the product heads could make big incorrect assumptions, that once you've made investments are hard to change later. I've seen both of these happen.

There are also dangers of lingering here too long. Imagine you've got one person who's very skilled at his or her job, and then leaves the company, maybe they retire. And all that organizational know how, walks right out the door with them. That can be a real problem. You have suddenly two weeks, you have no one's able to perform that process in the company. And then the one people more often think about at the individual contributor phase, is they just can't scale up the process to meet demand in time. You've got one person doing something, suddenly you need eight, and no one's even formalized the process. And this poor man or woman just can't answer the phone fast enough.

Image cc0 (prototype):
https://pixabay.com/en/ux-prototyping-design-webdesign-788002/

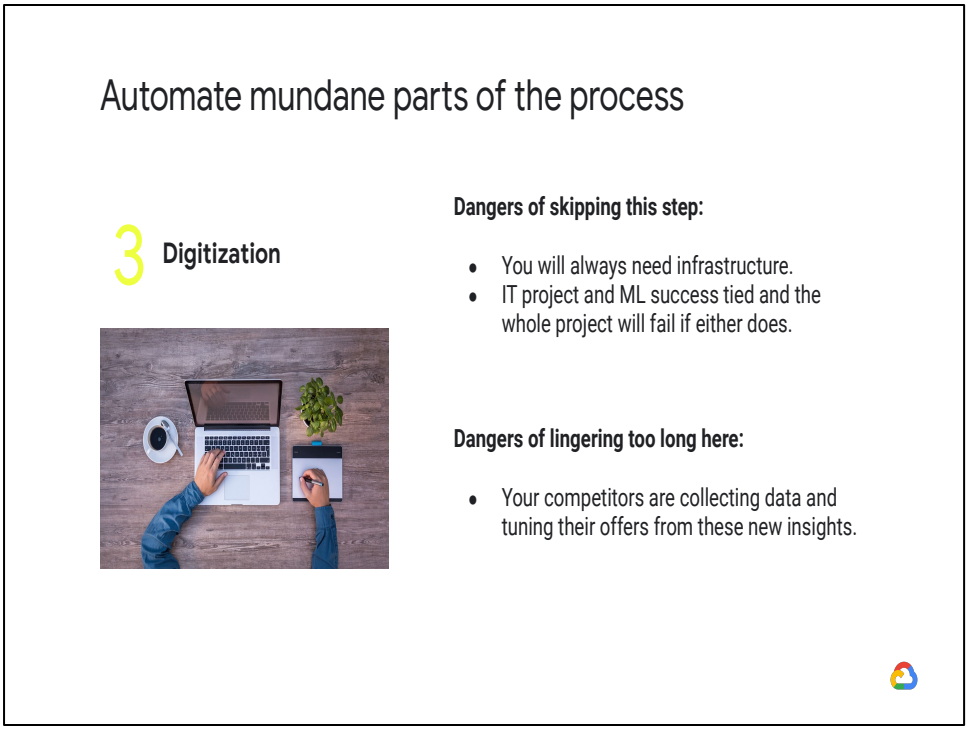
So what happens? Well, ideally, we move on to the delegation phase. And here, we're going to increase the number of employees working on some business process to hundreds or if you're Walmart or the US Military, even to millions. What's nice about this, is it allows us to gently ramp up the investment while we maintain most of the flexibility of the previous stage when we only had one person doing it.

The dangers of skipping this step. So if you skip this step, you're never forced to formalize the business process and to define success. Imagine, when we delegate. Imagine we're actually going to outsource, as an example. If you can't explain exactly what you need to happen in this business processes to someone 12 time zones away, you probably also can't explain it to a computer. Computers have even less context about what you're trying to accomplish than someone in India may. So this is a great halfway step to formalizing business processes. And also, getting organizational buy in for what is success, because this is not a cheap phase. There's also this great opportunity at this stage that human responses have an inherent diversity. And this becomes a test bed for great product learning. As each human, let's say as an example in a call center, each human answers the phone a little bit differently. And then, they have different customer outcomes. Then, in the organization, when you review that data, you can learn what's working and what's not. Then finally, great ML systems will need humans in the loop. And this is your opportunity to identify who you need. And inside Google, we have many ML systems. Many ML systems that generate a lot of value for the organization, for our customers, for our end users. And almost every single one has a team of people reviewing the algorithms, reviewing their responses, reviewing where they get confused, doing

random sub-samples. Because if your ML system is very, very important, then it needs to be reviewed and you're going to have people doing this. And you should think about ML as a way to expand the impact or to scale the impact of your people, not as a way of completely removing them. Because that's a very high bar for an ML system to meet.

What are the dangers of lingering too long here? I mean, the most obvious is, you're paying very high marginal cost to serve each user. When people answer phones, it's expensive. If you have someone sitting in a bank to wait someone to walk into the bank to open a mortgage, it's expensive. But we all know this. Perhaps a little bit more subtly, the more people you have in your organization, the more voices you have to say, automation is impossible. And this creates a kind of organizational lock-in, when you have so many stakeholders. Because the people have their own managers, their own H.R. reps, and so we want to win a delegate, but we want to always be moving forward. We want to always be going down this path whenever appropriate.

Image cc0 (delegation):
https://pixabay.com/en/people-girls-women-students-2557396/



And that leads us into the digitization phase, which is just a fancy term for getting computer systems to perform the mundane or repetitive parts of the process. And in general, the digitization is a part of automation. And in general, we think of automation economically as a way to trade upfront investment for a lower marginal cost or run rate. Now, automation gets us a lot of thing and maybe you first think about, Oh. Wait, we don't want to get rid of people. You're right. We want to scale their impact. We want to give our users a higher quality of service. And automation is a great way to do that. But because it involves so much upfront investment, it comes with its own risk.

So the dangers of skipping this step. Even with a great machine learning algorithm, you will need all the infrastructure of this step to be able to serve your ML at scale. Remember, machine learning perform some core, almost numerical task but it doesn't serve that task in a website or it doesn't have unit tests of its own. And all of this stuff that comes with software, you are also going to need it for ML. And most just going to replace a little piece of your otherwise very large software stack. Also, if you skip this step, and I see many people try, you start to untangle an. IT project, which we may call software, with an ML project. And then suddenly, either one of them fails, the whole project fails. And organizationally, it's very tricky to say what really happened. And you may find that management is dis-incentivized in the future from investing, because it's so hard to untangle what's going on. So I always encourage people to try to do as much IT project, show its own success, and its own milestones, and then see ML as icing on the cake.

The dangers of staying here too long. So, the other members of your industry are

collecting data and they're tuning their offers, they're tuning their operation from these new insights. And this is giving them an edge. And this is giving them, this is constructing a very positive feedback loop for them.

Image cc0 (computer):
https://pixabay.com/en/laptop-mockup-graphics-tablet-2838921/

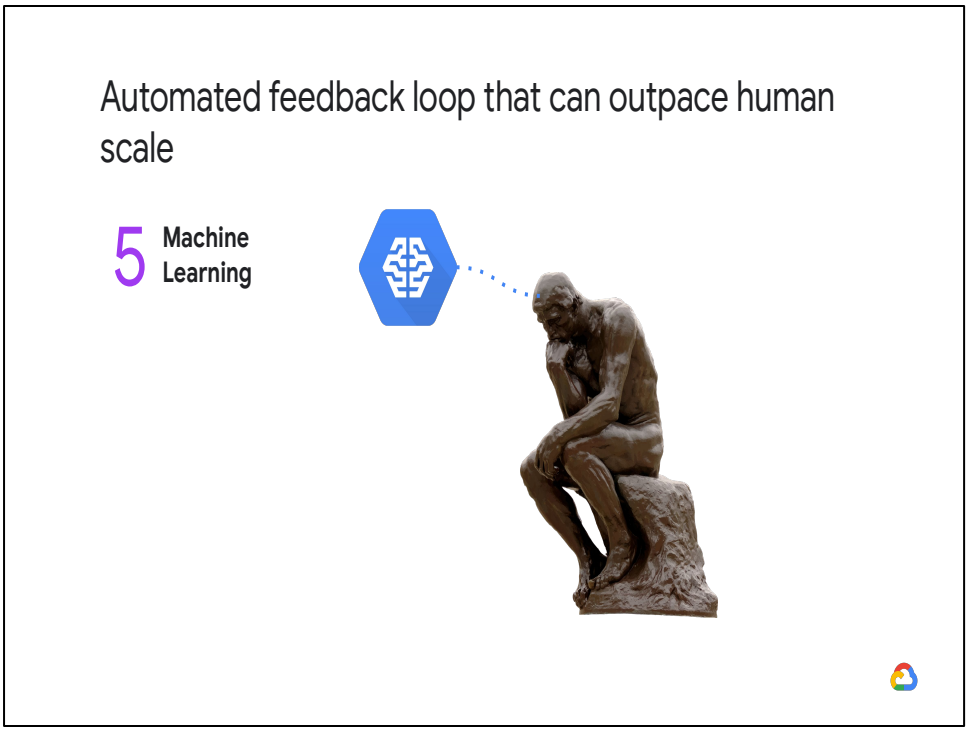
After digitization, we start to think about big data and analytics. And what we're going to do here is measure everything about your internal operations and external users. We're going to make it easy to review, summarize, deep dive, and this is a great moment to pause and reiterate your definitions of success, and then detune software algorithms that we described in the previous step. So why is this such a great opportunity to redefine success? So kind of the first stage, the initial individual contributor, all the way up to here, we've been using kind of like market research or kind of standard ways to hypothesize about what our users wants or wherever possible to apply data. But now that we're in the Big Data phase, we've automated the core process with digitization, which means that we may now extract an incredible amount of data because everything is digital for us. And with this new data, we can reassess our original definition and say, "Hey, we thought users wanted A, is that really true? Are we actually serving A correctly?" And this gives us many new insights.

The dangers of skipping the step. So if you try to go from software right into ML without ever having this kind of manual insight generation phase, you're going to run into some obstacles. First, you won't be able train your ML algorithm, because your data is not clean and organized. If you cannot make a histogram of your data, your algorithm cannot effectively make that histogram either. And more or less, that's what your algorithm is doing, is it's making many, many plots and performing regression on them. So if you can't make that chart, neither can your algorithm. Also, if you skipped this step, then you won't build a measure of success. So it's hard to tell if your ML algorithm's actually improving things. So we're going to start by manually doing all these things and then when we're ready, ML will automatically be cleaning, and all be

automatically acting on this clean and organized data. What happens if we stay here too long?

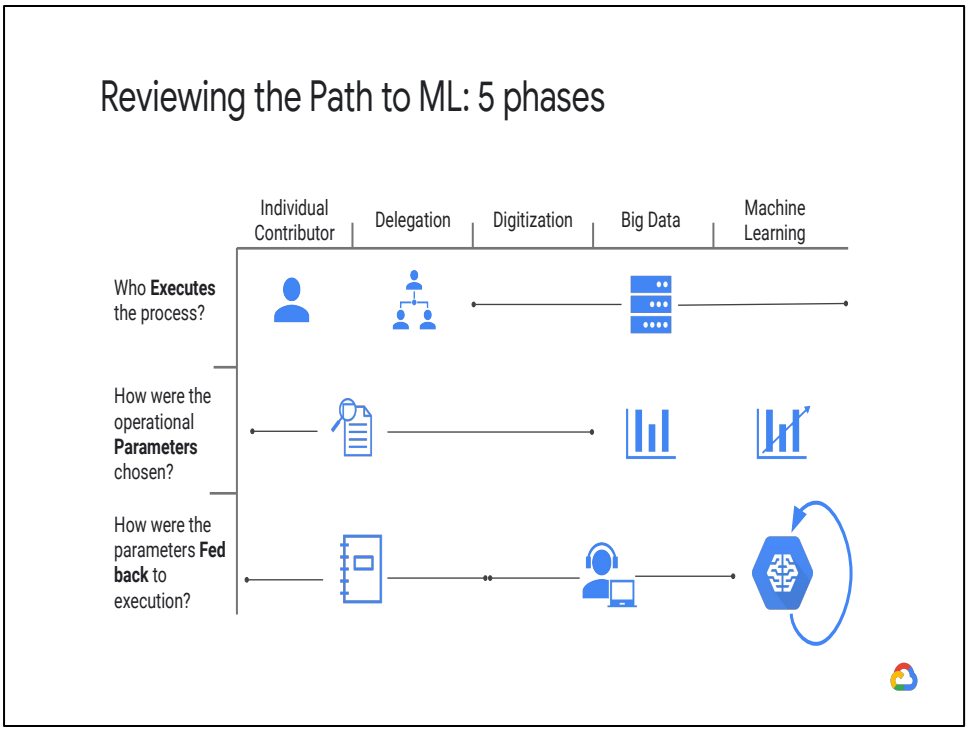
Actually, the dangers of that are not as major as some of the other steps. As an example, Google search stayed here for years. Everyone thinks that Google search is like this really maniacally impressive ML, natural language processing algorithm. And maybe it is now. But it wasn't for many, many years. It was actually a hand-tuned algorithm. And this is actually really interesting. You should take this message to heart that like you can get very, very far without handing over everything to ML. But of course, without ML, you are going to limit the complexity of problems you can solve. And the speed at which you can solve them.

Image cc0 (analytics):
https://pixabay.com/en/analysis-analytics-business-chart-1841158/



So finally, let's talk about the last stage, the machine learning phase. Here, what we're going to do, is we're going to complete that feedback loop we talked about before. We're going to automate each of those blocks between measuring success and tuning the software algorithm. Eventually, this will outpaced humans ability to handle the number of inputs and the corner cases involved in your real world data, in your real world use case. Generally, roughly at Google, we expect about a 10 percent key performance indicator increase in Google products on top of all the human-hand tuning, just from ML's ability to handle each of those little details so incredibly well. And of course, the reason we're all here, what makes machine learning so fascinating is we're going to escape the limitations of human cognition in solving our business problem. We're going to get faster answers, more nuanced treatment of details, and perhaps what is most mind blowing to me, is there will be one brain learning from billions of interactions every day. Google search as an example from the last slide, is able to watch how one user is searching for new terms and apply that to searches happening for another user as it learns about the content to the world around it. And this is truly fascinating. In your organization, this will happen to. As you have one ML algorithm that is learning from many disparate interactions around it.

Image cc0 (Thinker):
https://pixabay.com/en/art-auguste-rodin-bronze-famous-1301872/

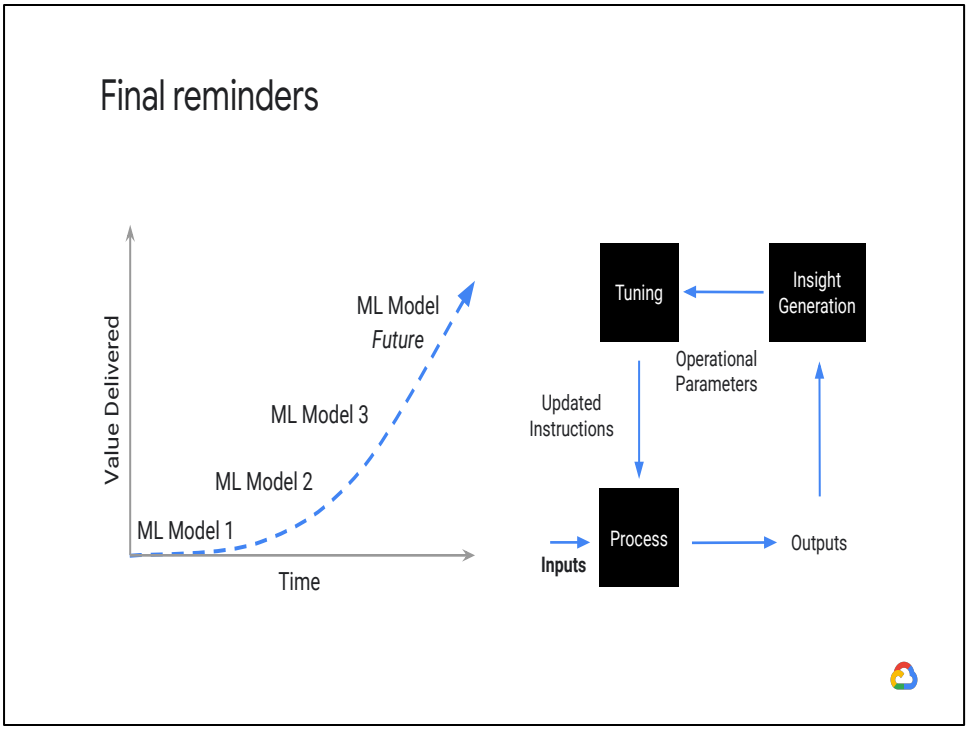


So, we talked about these five phases and it was contributor, delegation, digitization, big data, and machine learning. One of the axes of change between each of these is who or what is executing the process. So, as we look at this diagram, which was made for you visual learners out there, we see that in the individual contributor phase, we have one person, when we get to delegation, we have many people. But the other three phases, it's always the same entity executing the process. It's always a computer or some server rack.

But there's another axis of differentiation, and that's how do we choose the operational parameters? So, from the individual contributor, all the way out to digitization, we're doing some kind of market research or some kind of less than pure, we call, more hypothetical exercise about, well, this is what I think users want, this is what I think we should try to accomplish. And then we get to big data. We actually, for the first time, have the opportunity to produce an incredible amount of very deep rich market insight or user insight or operational insight. And this gives us a new method to choose these operational parameters. And then in machine learning, you could imagine we're just going to start to regress or to make some, we're going to fit the data with a line or obviously maybe more complicated curve, where we start to actually perform these fitting parameters and extract to cases we may not have seen yet.

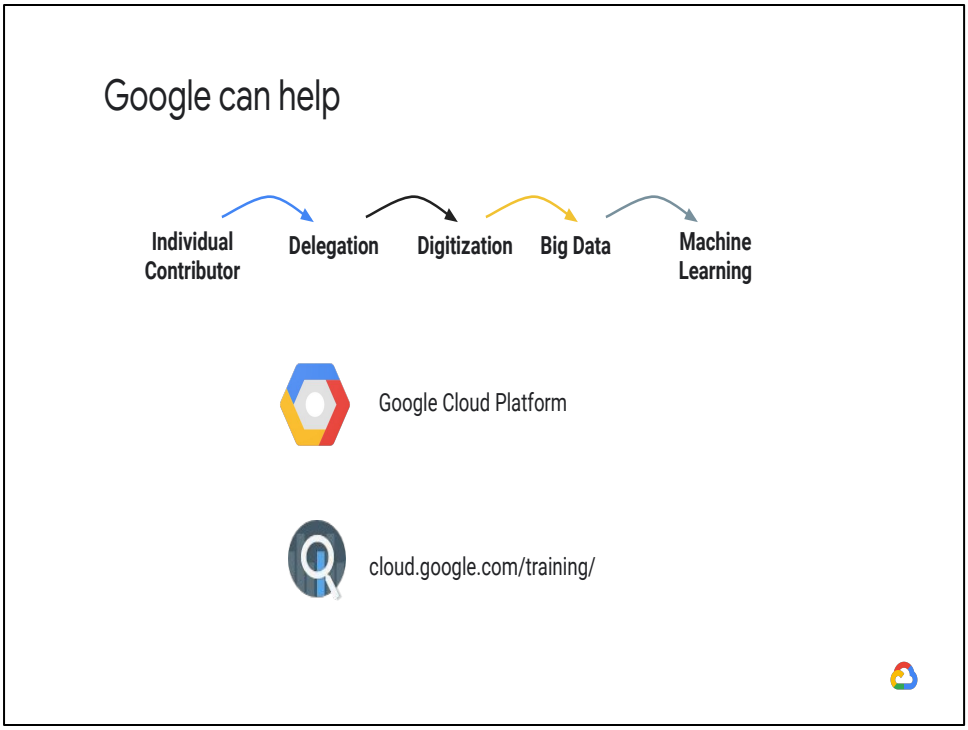
And then this final axis of differentiation is given that we have these better operational parameters, how are they fed back into the execution of the core process? So, as long as we of humans accomplishing the task in the first row, then the way we feed

back the execution, is we have some kind of like H.R training or reference manual or something that the humans refer back to as needed. Of course, if you have a truly individual contributor, the worst case scenario is nothing's written down. So, we always hope that there is some kind of employee handbook. But then, as we move from digitization and big data, now, when we have these new operational parameters, we know what we should be offering to our users. We're going to rely on our software engineer with the headphones to go ahead and put that back into our code so that it may be acted on and presented to our users. And then, when we get to machine learning, machine learning is actually going to be deriving these new operational parameters and feeding them back in automatically.

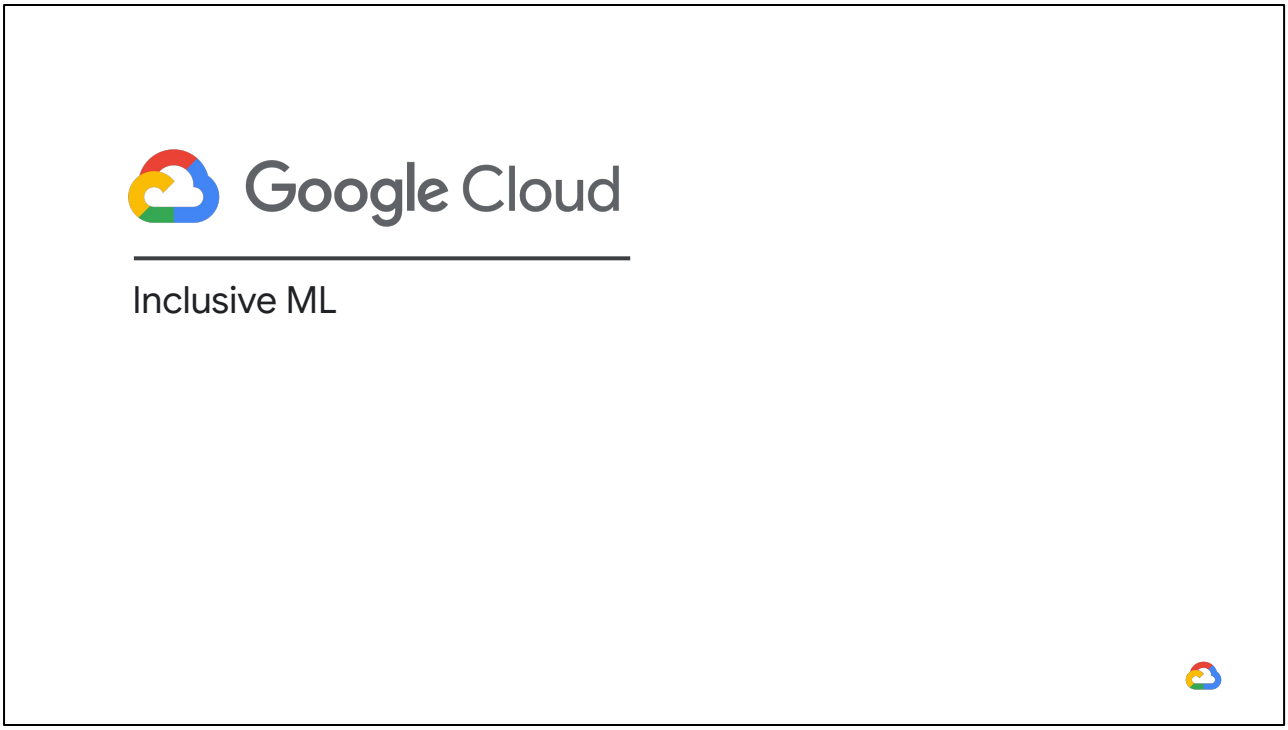


So, I want to zoom even further back into some final reminders about this lecture in general. It's tempting to jump from nothing to a fully machine learned solution, but it's a risky move. Success in Google and likely in your organization too, typically follows a more structured approach that steadily increases that upfront investment as the business uncertainties decrease with more experience and research into your users and your product niche.

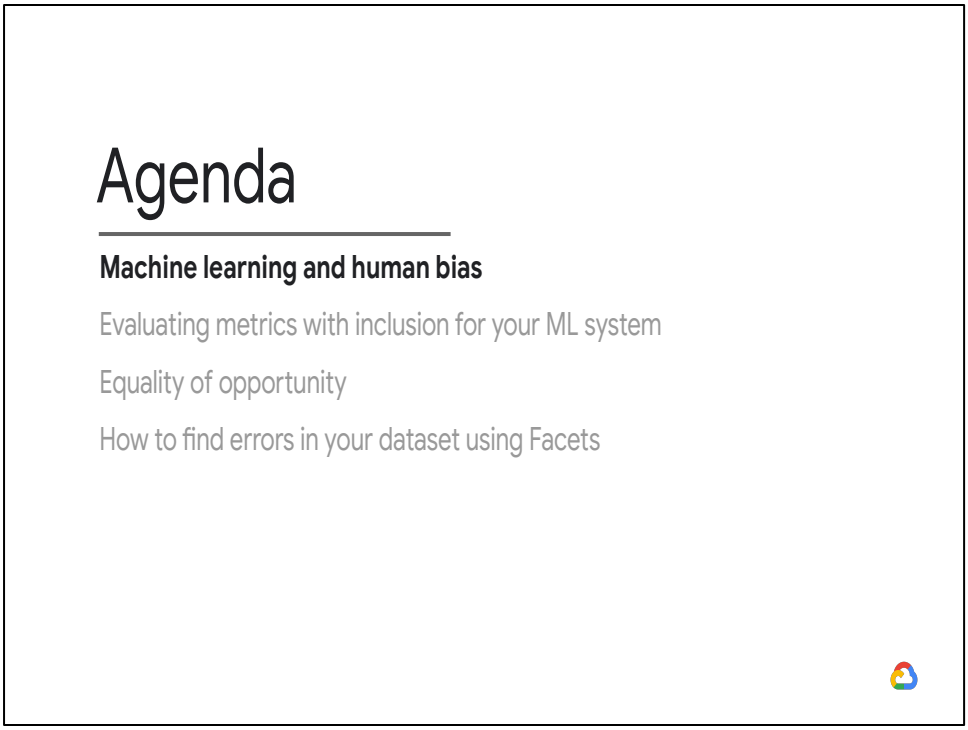
And we've talked a lot about this business process score feedback loop here. And the main goal of the Path ML is of course, to automate all the blue boxes, and that's what we're trying to do and that's the Path ML.



The good news is Google can help, Google has a variety of services that can help you in across this path, starting an innovation contributor, but of course, because we're cloud offering, we get very strong once you get to the digitization and beyond.

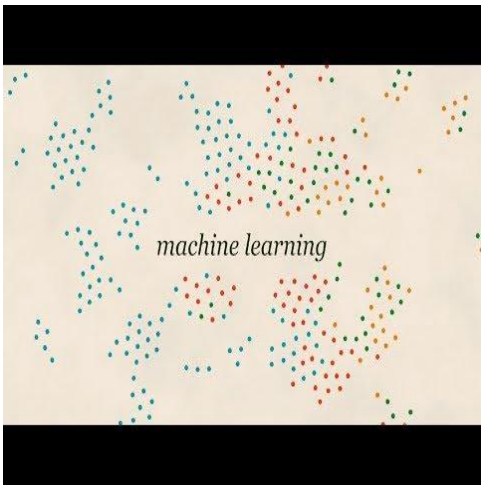


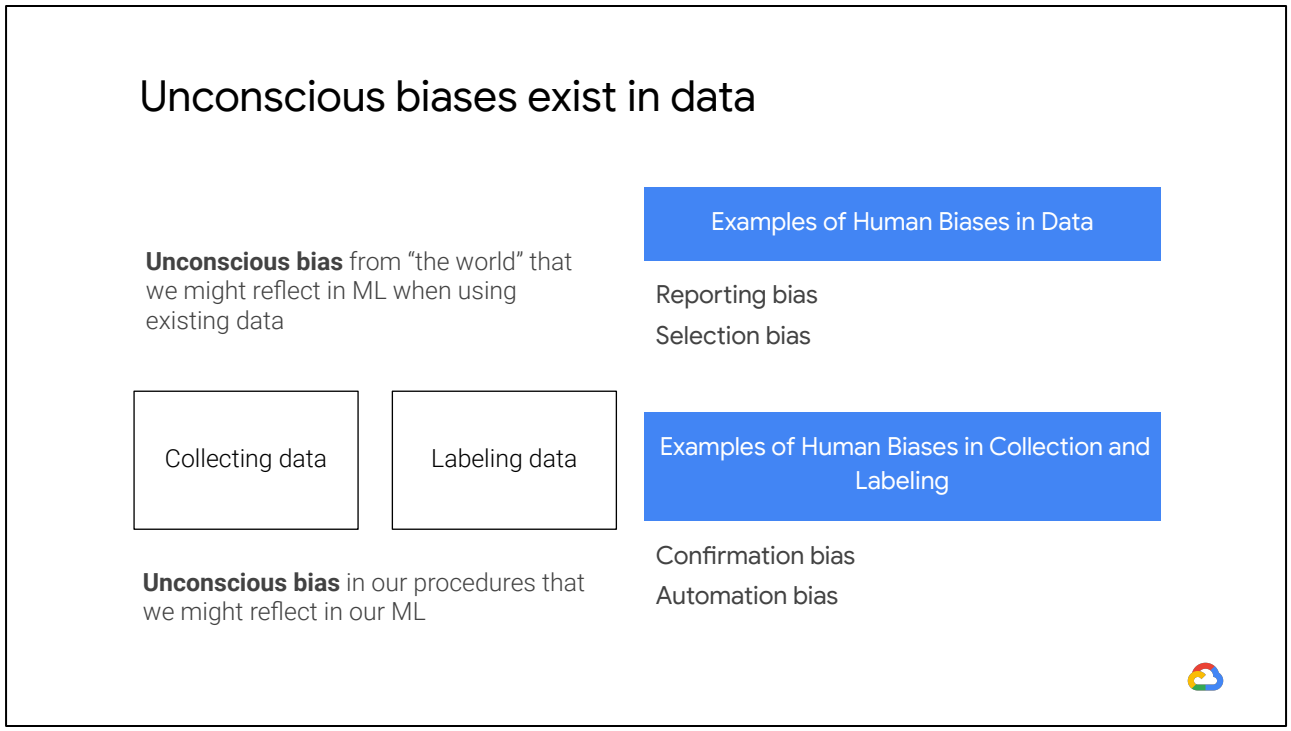
Welcome to ‘Inclusive ML’.



You'll first watch a video that explains where bias in machine learning models originates, and the importance of building inclusive machine learning systems.
After the video, you'll walk through some of the ways in which you can understand the tradeoffs a machine learning system makes—and how these trade-offs map to evaluation metrics that you can compute.
From there, equality of opportunity will be introduced, a methodology that builds on top of these evaluation metrics in order to achieve a more desirable outcome—an outcome where there is an equal chance of a machine learning system correctly classifying an outcome, irrespective of any sensitive attributes.
And finally, as you know, machine learning systems are fueled by data. So getting the best results out of a machine learning system requires that you truly understand your data—and that holds true for making machine learning systems inclusive. In the last section, you showcase an open source visualization tool for machine learning data called Facets, which helps you explore the intricacies of your data set—and provide some suggestions on what to look for when assessing the inclusiveness of your training data.

Human biases lead to biases in ML models





Unconscious biases exist in our data. They exist in two forms:

First, there are the human biases that exist in data because data found in "the world" has existing biases with regard to properties like gender, race, and sexual orientation. We can think of those biases in terms of the way they manifest in our data samples.

There may be *reporting bias* by our subjects because they only choose to reveal certain aspects about themselves or their opinions. There may be *selection bias*; that is, those subjects that get into our samples only represent a privileged type of user.

There may be *selection bias*; that is, those subjects that get into our samples only represent a privileged type of user.

Let's say that we are developing an ML model to find fraudulent transactions. Can you see what kinds of problems we might have?
Suppose we train the model on historical transactions. What could go wrong there? Well … we may have a reporting bias if our customers tend to use cash in grocery stores. We only know about the places where they tend to shop with the card, and we might wrongly assume that all grocery store transactions are fraudulent. Just because we have not seen enough grocery store transactions that use a card. However, many US states issue state benefits in the form of debit cards and this will have the impact of denying poor users the ability to use those cards for food. This is an example of reporting bias that caused a real-world outcome.

Or …

Maybe our cards get used at only large department stores. Our model might start to reject transactions at smaller stores because there aren't enough of those. That's an example of selection bias from the perspective of the small business owner who doesn't get any business from the credit cards we are issuing.

Second, we can also run into human biases which arise as part of our data collection and labeling procedures.

Confirmation bias refer to only looking for data which may confirm our hypotheses.

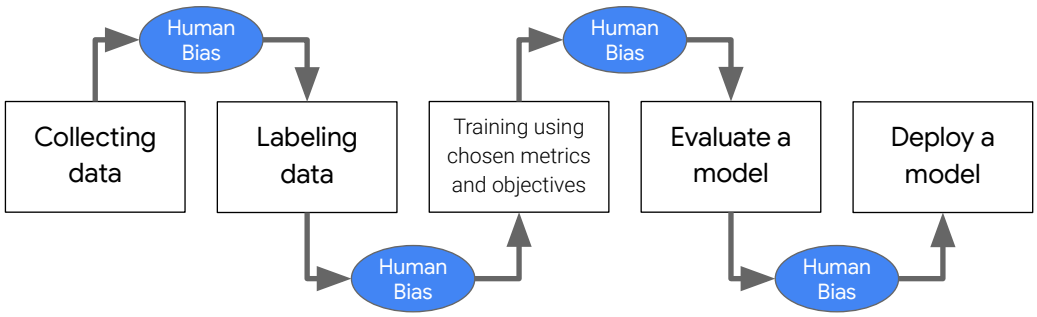
And *automation bias* refers to the biases which crop up when the data we use is solely the data which is easily automatable.

For example, suppose we are building a product recommendation system.
We believe that someone who buys a tie will look for men's shirts. So, we might restrict our data collection to only men's apparel items.
But lots of women do wear ties. This is an example of confirmation bias.

Automation bias is surprisingly common. If men's apparel and women's apparel are run by different parts of our organization, it is quite possible that this is why we might have collected only men's shirt sales. The presence of a silo has caused unconscious bias.
You might also have automation bias if you have a fancy digital data collection system at some locations, but other locations are still sending you faxes.

Chances are that you will prioritize the data that is easy to collect.

A typical ML pipeline *with bias*



So, what's the impact of biases in collecting data and labeling data?
It affects the entire pipeline!

In machine learning we begin with our data; the biases in the original data are going to be reflected downstream in our models and consequently are going to result in potentially biased outcomes.

The result is that biases can appear in at every point of the data pipeline. It appears at the point of collection. What kinds of data are accessible? Does the availability of data privilege a particular group of people? Does it put another group at a disadvantage? What transformations are you making to your data that may exclude one group compared to another?

It can also appear in the process of labelling. What human biases are human annotators introducing into the data? How stable are your categories of classification?

It can occur at the model level. Does a particular objective put certain subgroups of people at a disadvantage compared to the group in aggregate?

And those biases will appear downstream in the output of our model, and our users will see an effect or outcome as a result.

2 Avoid creating or reinforcing unfair bias

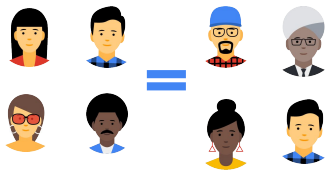
ML models learn from existing data collected from the real world, and so an accurate model may learn or even amplify problematic pre-existing biases in the data based on race, gender, religion, or other characteristics.

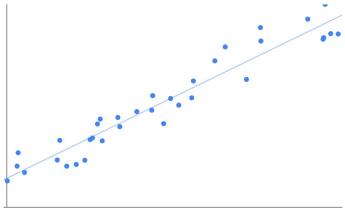
ai.google/principles

We recognize that such powerful technology raises equally powerful questions about its use. How AI is developed and used will have a significant impact on society for many years to come. As a leader in AI, we feel a deep responsibility to get this right. So Google has announced seven AI principles to guide our work going forward. These are not theoretical concepts; they are concrete standards that will actively govern our research and product development and will impact our business decisions.

The challenge we run into when creating a system that is fair and inclusive to all is that ML models learn from *existing* data collected from the real world, so an accurate model may learn or even amplify problematic pre-existing biases in the data based on race, gender, religion, or other characteristics.

A Checklist for Bias-Related Issues





Here's a quick checklist for situations where you should watch out for bias-related issues.

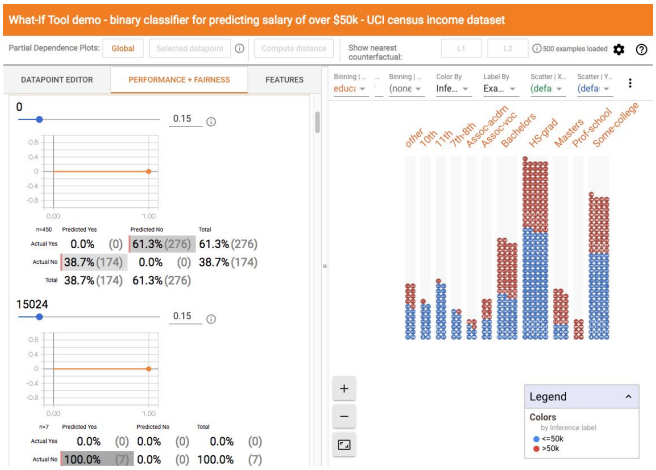
Does your use case or product specifically use any of the following data: biometrics, race, skin color, religion, sexual orientation, socioeconomic status, income, country, location, health, language, or dialect?

Does your use case or product use data that is likely to be highly correlated with any of the personal characteristics listed above (for example, zip code or other geospatial data is often correlated with socioeconomic status and/or income; image/video data can reveal information about race, gender, and age)?

Could your use case or product negatively affect individuals' economic or other important life opportunities?

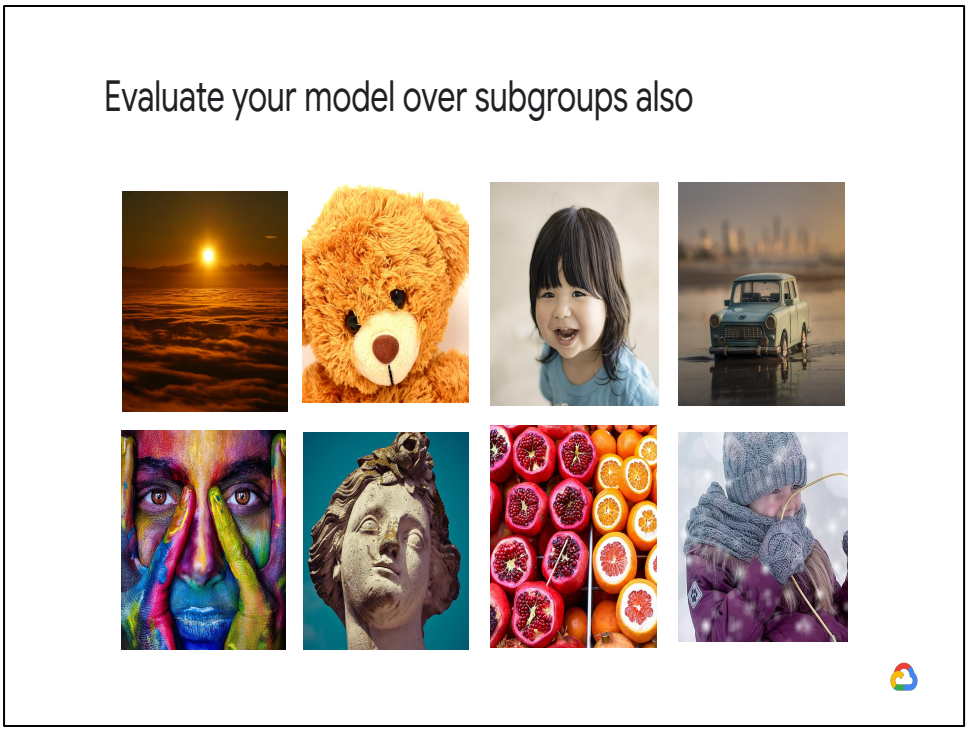
https://cloud.google.com/inclusive-ml/#fairness-in-ml-automl

Tools for Responsible AI



There are tools available from Google that help you diagnose fairness issues in your data, in your labels, and in the effects of predictions.
If you are implementing models, please use them.

This is the What-If tool. It's available for free and you will be able to access it from within Tensorboard. It's designed to let you: visualize inference results, Edit a datapoint and see how your model performs, Explore the effects of a single feature, Arrange examples by similarity, View confusion matrices and other metrics and Test algorithmic fairness constraints.



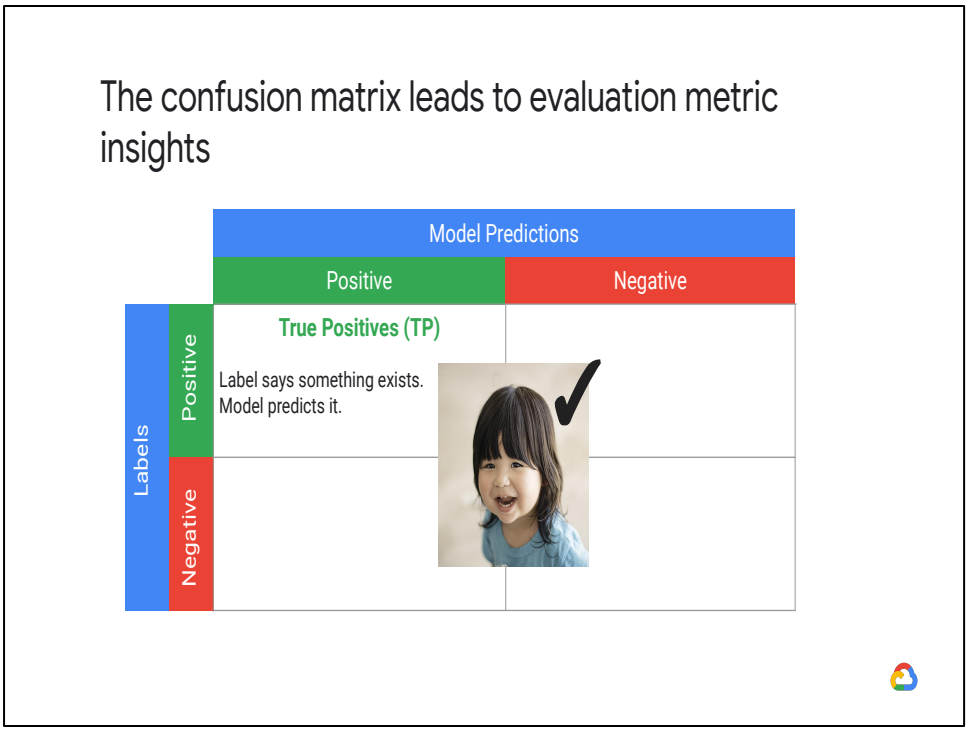
One of the key things to really nail, which will help in understanding inclusion and how to introduce inclusion across different kinds of groups across your data is by understanding the confusion matrix.

While you may be familiar with evaluating your model over the entire dataset, you are talking about something slightly different here. You are saying that instead of just looking at how your model performs overall, over your entire dataset, break the performance down to the subgroup that you wish to improve performance on to make your ML system more inclusive. For example, suppose you are doing face detection. Essentially, you are building a ML model to say whether or not there is a human face in the photograph. As the photographs on this slide indicate, this is not necessarily an easy problem.

Your subgroups might be men and women, adults and children, people with hair and people who are bald. You want to look at the performance of your model across all these subgroups to make sure that unconscious bias has not crept in.

https://pixabay.com/en/gibbon-wildlife-indonesia-mammal-3007235/ (cc0)
https://pixabay.com/en/art-man-male-head-face-myanmar-2991282/ (cc0)
https://pixabay.com/en/statue-sculpture-figure-1275469/ (cc0)
https://pixabay.com/en/girl-mejk-color-rainbow-model-2696947/ (cc0)
https://pixabay.com/en/teddy-soft-toy-bears-funny-2989139/ (cc0)
https://pixabay.com/en/doll-dress-colors-beautiful-1907768/ (cc0)

https://pixabay.com/en/sunrise-selva-marine-clouds-1959227/ (cc0)
https://pixabay.com/en/miniature-car-model-toy-automobile-1802333/ (cc0)
https://pixabay.com/en/market-fruit-oranges-pomegranates-3016769/ (cc0)
https://pixabay.com/en/baby-smiling-child-cute-kid-2553539/ (cc0)
https://pixabay.com/en/christmas-girl-slide-person-2992358/ (cc0)



A common way that you evaluate performance in machine learning is with a confusion matrix. This is only for classification problems, and you use other methods for other types of problems, but for the purposes of this module, you will use a confusion matrix to explain the points.
The idea in using the confusion matrix in order to look at inclusion is to: create a confusion matrix for every subgroup in your data where you're interested in measuring performance.

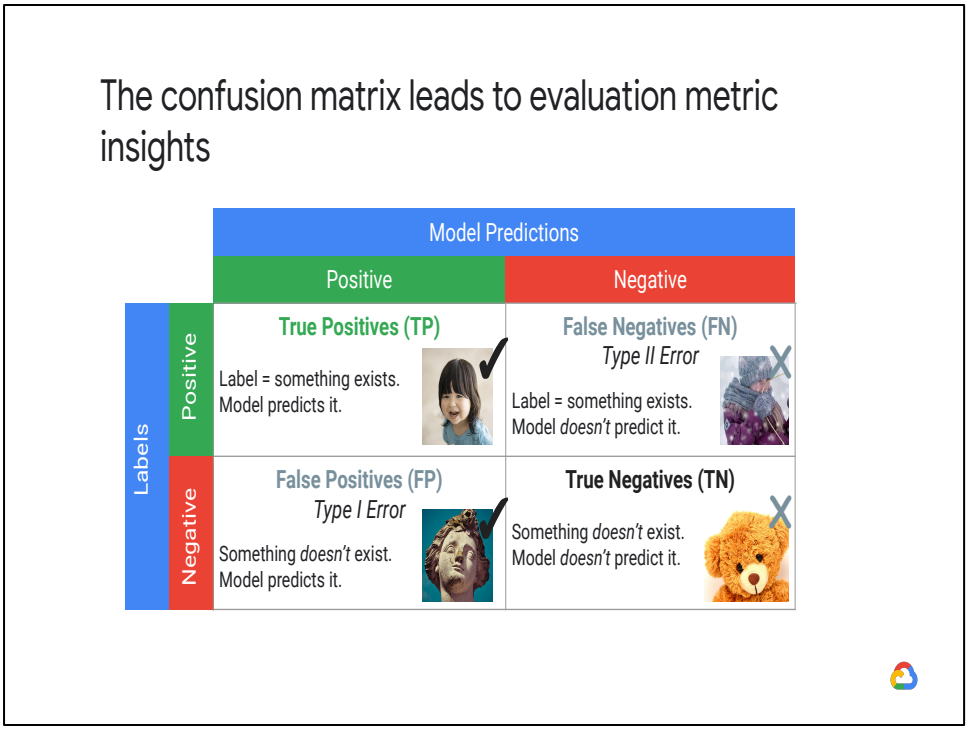
In the confusion matrix, you have comparisons between your labels (which may or may not necessarily reflect your ground truth—because sometimes we don't necessarily have access to ground truth)—and you compare those labels to your model's predictions.

From here, you look at the positives and negatives. So in your labels, there are some things that are there in the data—and that would be considered correct, or a positive label. And there are other things that are not there in the data—and you considered those incorrect, or a negative label.

On the ML side in the predictions, you have positive predictions about what's there—and you have predictions that are not there, which are negative.

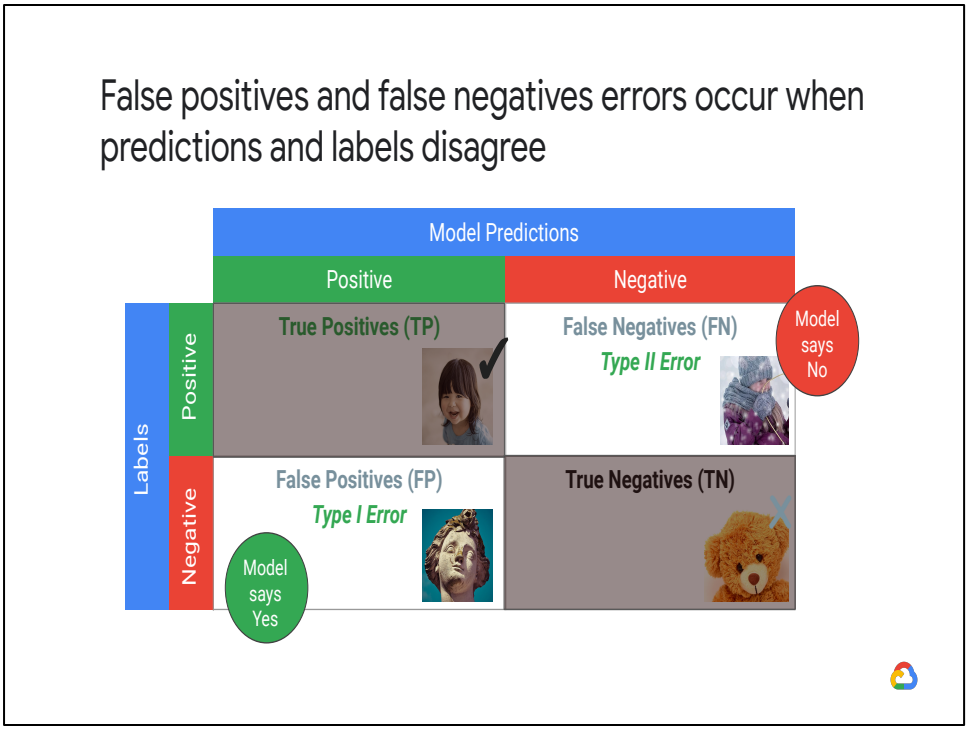
You compare these in the confusion matrix in order to start to tease out the errors a ML system is making — starting with the true positives, which is when the label says something is there and the model predicts it.

So, in the case of face detection, a true positive would be when the model accurately predicted that there was a face in the image.



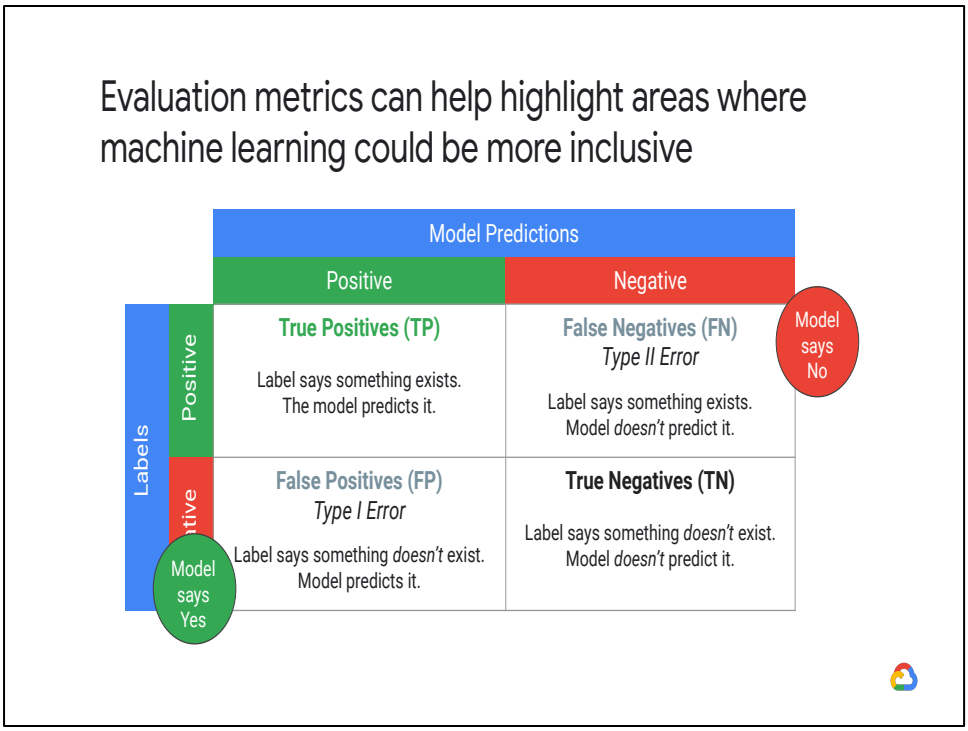
Now, when the label says something exists and the model doesn't predict it—that's a false negative. So, in the face detection example, the model says that there is no face in the image—when the image's label says there *is* a face.

And similarly a false positive. The label says there is no face, but the model finds a face. Perhaps there is a statue in the image and the model falsely identifies it as a face.



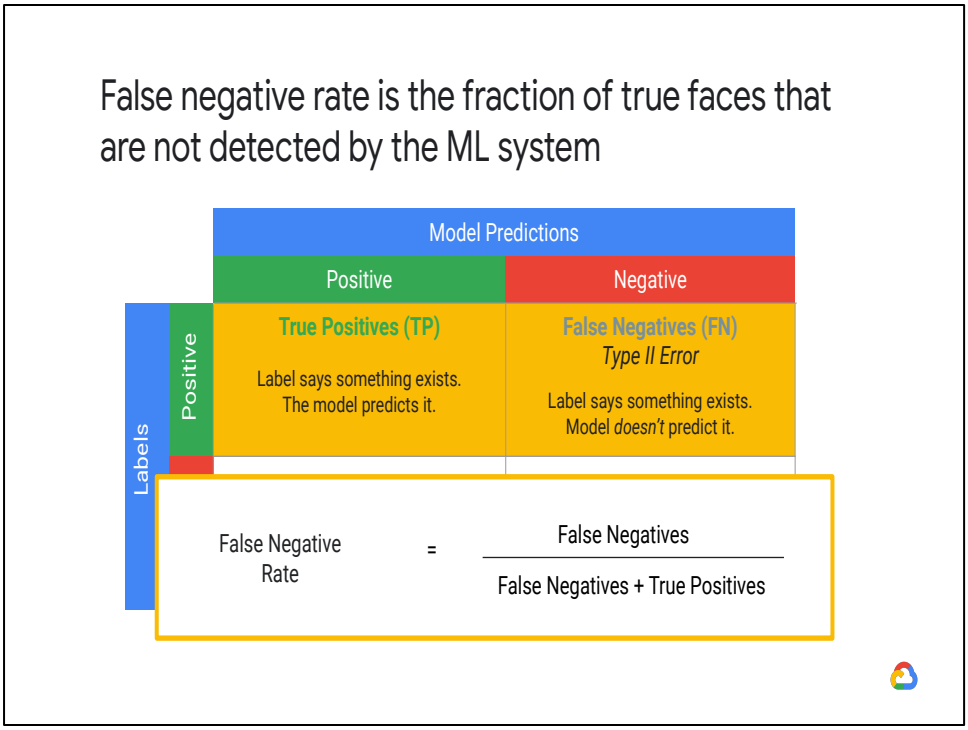
But really what you want you to focus in on here are the false negatives and false positives. Remember: false negatives are things that you incorrectly do not predict; things you exclude when instead it should have been included. And false positives are things that you incorrectly predict; things you include that aren't actually there in the label and should have instead been excluded. And these are also called Type I errors and Type II errors in other domains.

The cool thing about this sort of basic breakdown into these four different kinds of matches to the labels is that you can start to calculate a ton of different metrics that can be used to gauge the amount of inclusiveness in your model.

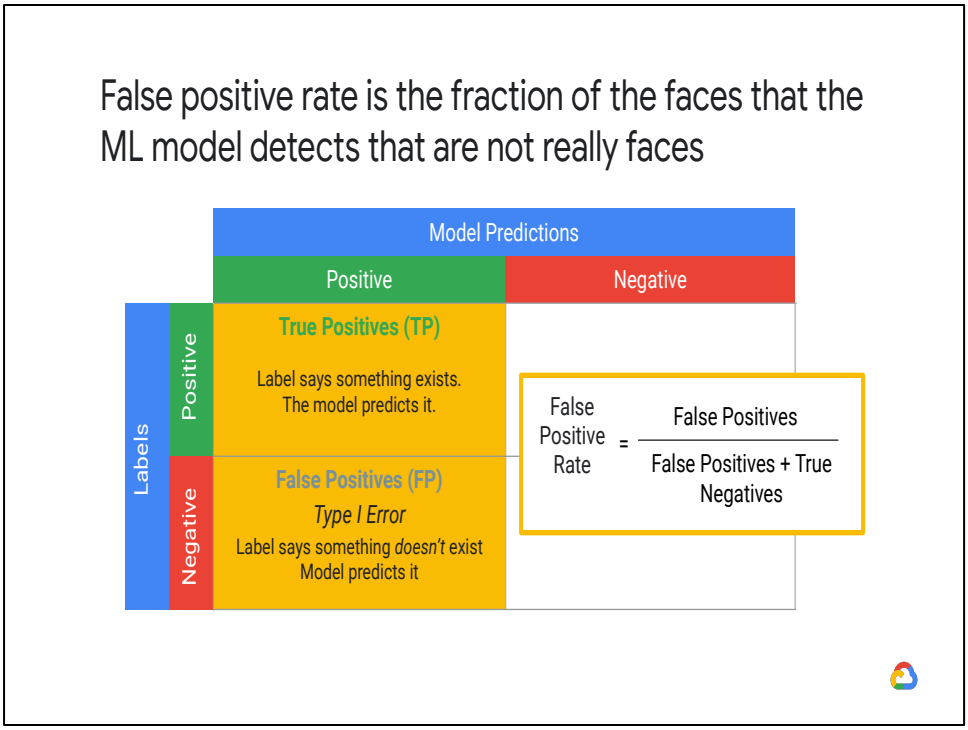


So now that you have this confusion matrix set up, you can start to calculate all kinds of evaluation metrics that could help you identify areas where a machine learning system could be more inclusive.

But with respect to making ML more inclusive, you tend to really focus heavily on false positive rate as well as the false negative rate in order to get a sense of on how adversely affected a subgroup might be performing.



You can calculate things like the true positive rate, sensitivity, or recall for example—all of which represent the proportion of times your model predicts, say, a face in an image when the label itself also shows there being a face in the image. All you need here are the corresponding true positives and false negatives values in order to calculate the recall.



Another example of the sort of derivations you can calculate from a confusion matrix are things like the precision, which represents the proportion of times when the model predicts the labels correctly—factoring in both when it's a positive label (there is a face in the image) and the model predicts it's the positive label, as well as when it's a negative label (there isn't a face in the image) and the model predicts it's the negative label.

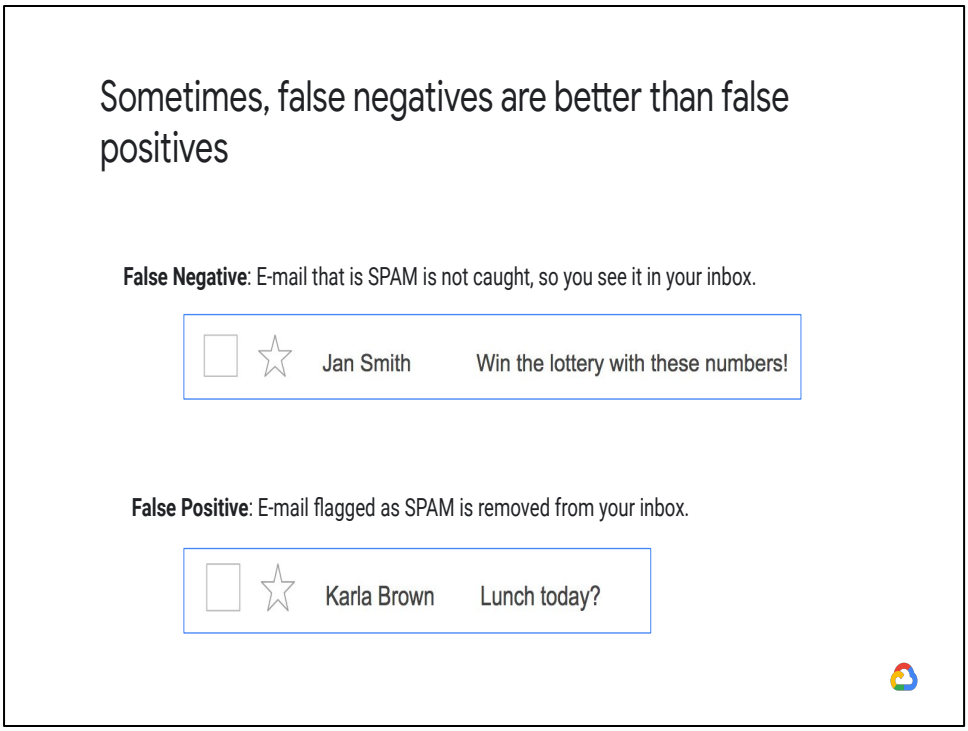
In this calculation, all you need are the corresponding true positives and false positives measurements.

False positive rate, False negative rate, True positive rate, precision, recall. So many metrics!
How should you go about selecting which metrics to focus on for the purposes of making your ML system more inclusive?

The answer is that it depends. It depends on the outcomes of your false positives and false negatives. Depending on the trade-offs between the two, perhaps you may want your machine learning model to have low recall (missing a lot of stuff) in exchange for high precision (or of the limited amount of stuff the ML classifies, it's all correct).

Take this example of a machine learning model that's determining whether or not an image should be blurred to preserve privacy. A false positive would result in something that doesn't need to be blurred but gets blurred anyway because the model predicted that it should—which can be a bummer. But a false negative here is something that needs to be blurred but is not because the model doesn't predict that it should—and that could result in identity theft because the privacy of the individual in the image was not blurred.

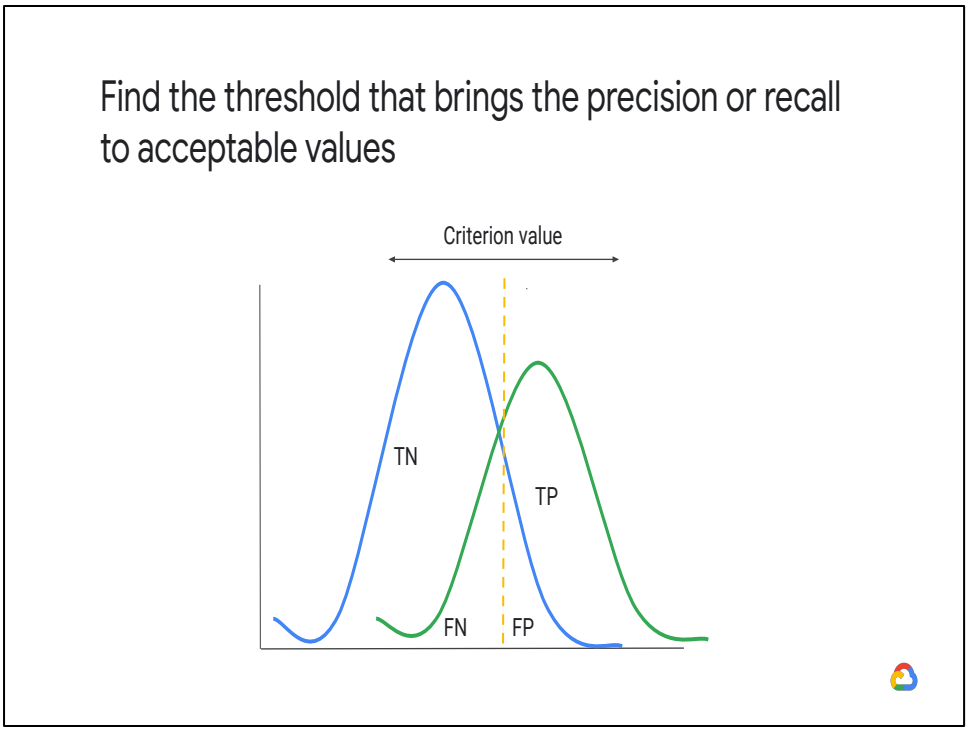
So in this example, you may want to minimize as much false negatives as possible, so you would focus your metrics around achieving a low false negative rate.



On the flip side, you might have situations where it might be better to encounter a false negative than a false positive. Let's say you are working on a spam filtering model. A false negative would result in a spam message not getting caught by the model—so you end up seeing it in your inbox and can be an annoying experience. But what happens when you encounter a false positive?

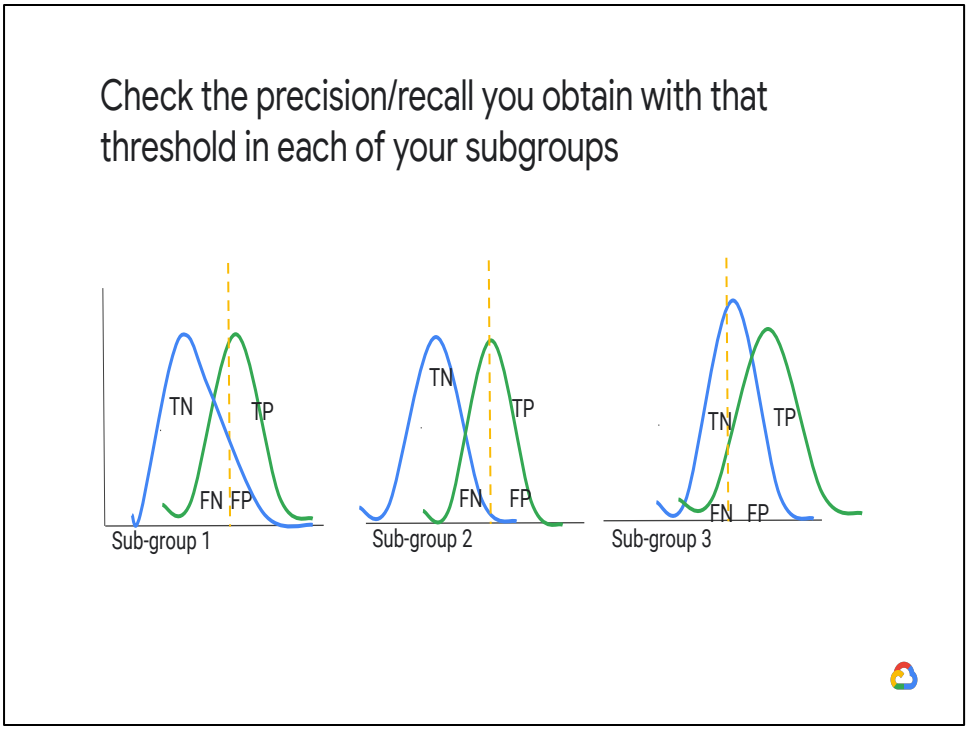
The result is that potentially a message from a friend or loved one gets marked as spam and removed from your inbox—which is a total loss!

So in this case, perhaps the metric to focus on here is reducing the false positive rate as much as possible.



First, use the evaluation metrics to pick a criterion value.

But once you figure out what the right set of evaluation metrics to focus on, make sure that you go one step further. Calculate those metrics in mind across the different subgroups within your data.

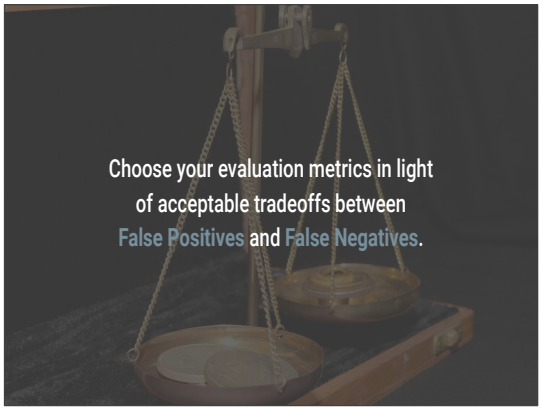


For example, you may find that a false negative rate @ 0.10 is acceptable for the problem you're trying to solve with your machine learning system. So now given that overall rate, how does that rate look across the subgroups.

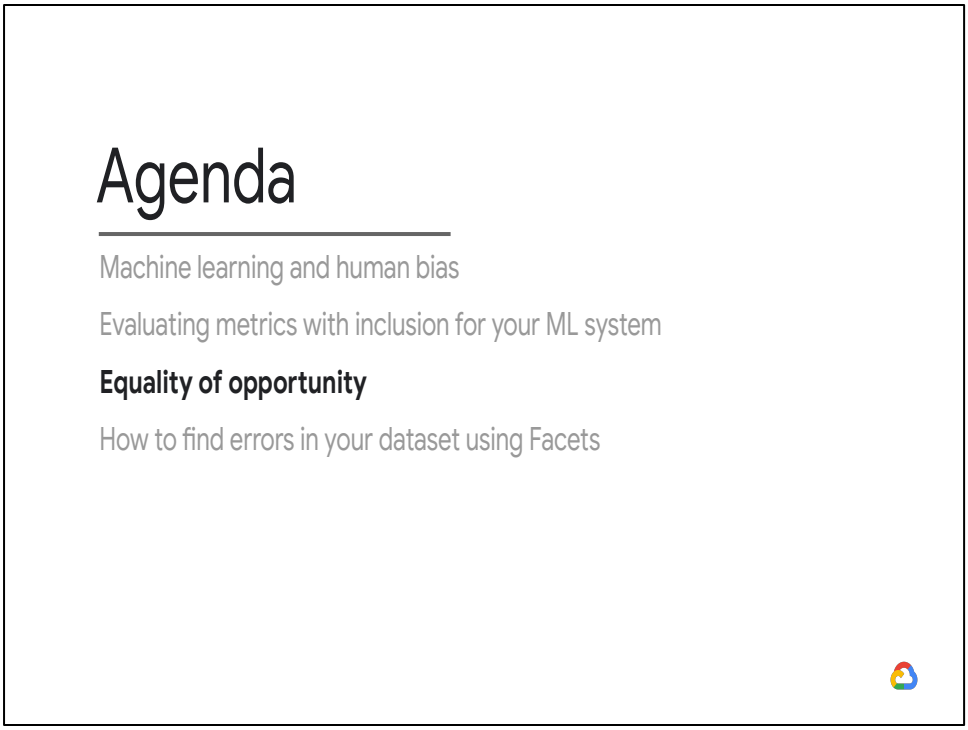
As shown in these plots, and after you compute your metrics, you can visualize the distribution of your evaluation metrics across each subgroup—as depicted by the blue and green distributions, each represent a separate subgroup within the data. But once all of that is in place, then it's just a matter of finding the point that is at an acceptable value and compare those values across the subgroups.

By incorporating these methodologies, you are one step closer identifying ways in which you could make your machine learning system more inclusive.

Evaluating metrics are some of the key things you can do to measure how inclusive an ML system is

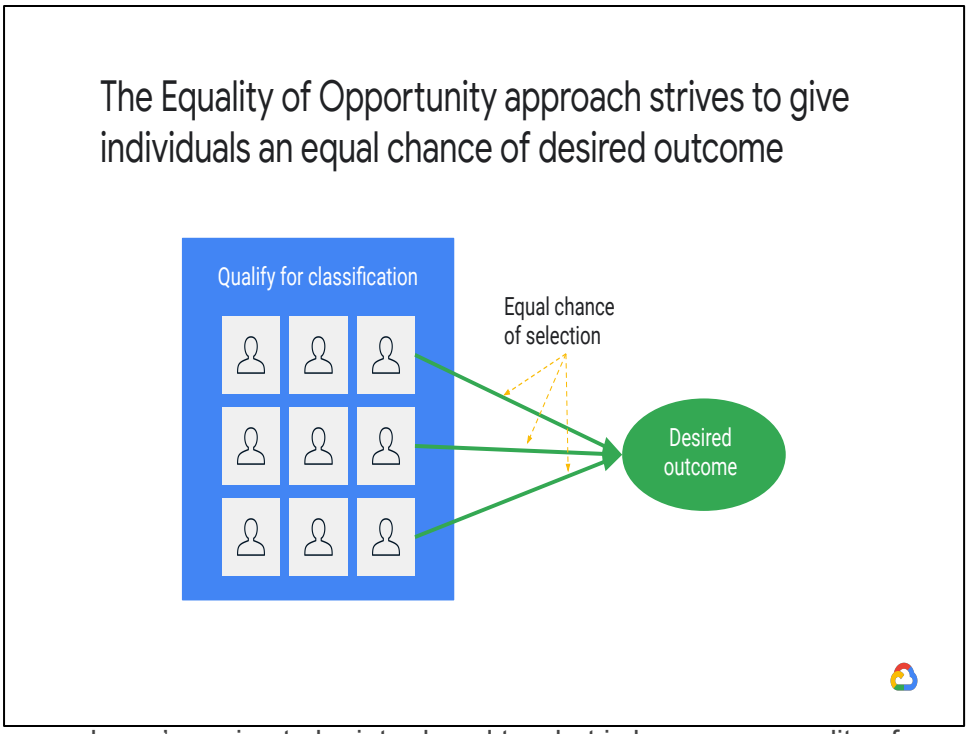


So to reiterate, evaluating metrics are some of the key things we can do to measure how inclusive a machine learning system is—and it's important to do so in light of the acceptable tradeoffs between your false positives and false negatives.



Your ML system will make mistakes. It's important to understand what these errors look like and how they might affect the user's experience that's driven by the output of your ML model.

In this module, you'll discuss some of the ways in which you can evaluate inclusion as you're developing and testing your machine learning model.



The approach you're going to be introduced to what is known as equality of opportunity, and it goes something like this:

Let's say you have a model that's intended to work across all users, irrespective of who they are or where they come from. Ideally, all users who qualify for a desirable outcome generated by your model should have an equal chance amongst all users of being correctly classified for that desirable outcome.

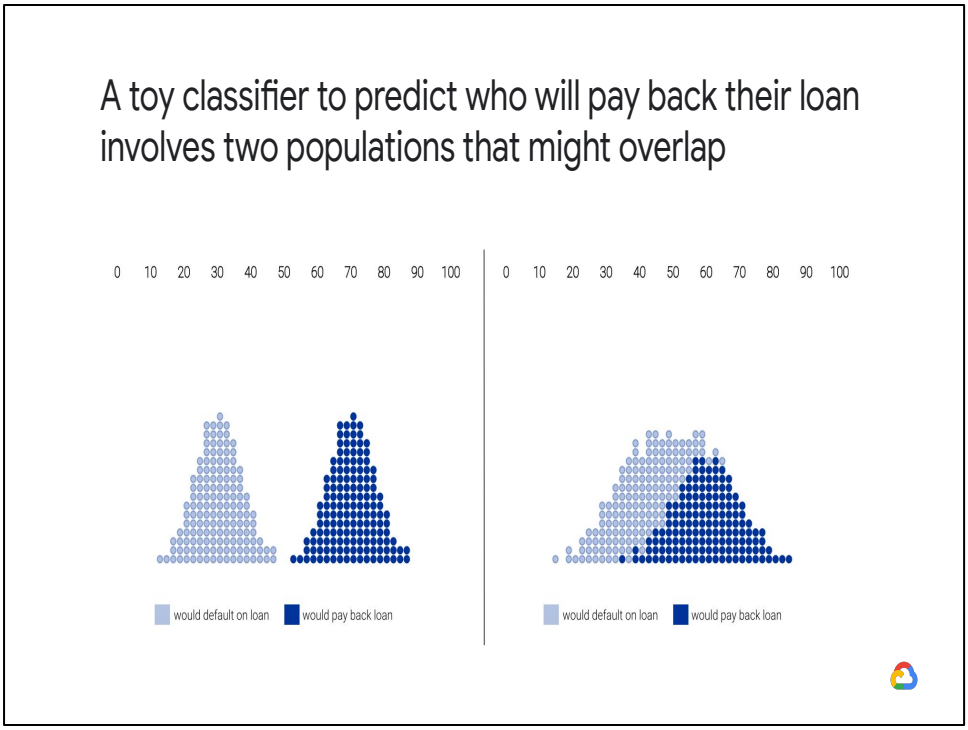
Let's say you are working for a bank and we are building a ML model to help determine whether or not to approve a loan. What does equality of opportunity mean in this context? Ideally, all users who qualify for a loan have an equal chance amongst all users of being correctly classified for that loan approval. In other words, the chances of a person being qualified for a loan should be the same regardless of which protected subgroup they are part of. We are saying that if you keep everything about a person the same, and change them from being a member of one subgroup to another, their chances of qualifying for the loan should remain the same.

So why should you incorporate such an approach into your machine learning system?

Because an approach like this gives you a way to scrutinize your model in order to discover possible areas of concerns. Once you identify opportunities for improvements, you can now make the necessary adjustments to strike a better tradeoff between accuracy and non-discrimination—which, in turn, could make your

machine learning model more inclusive.

Let’s see how to do this.

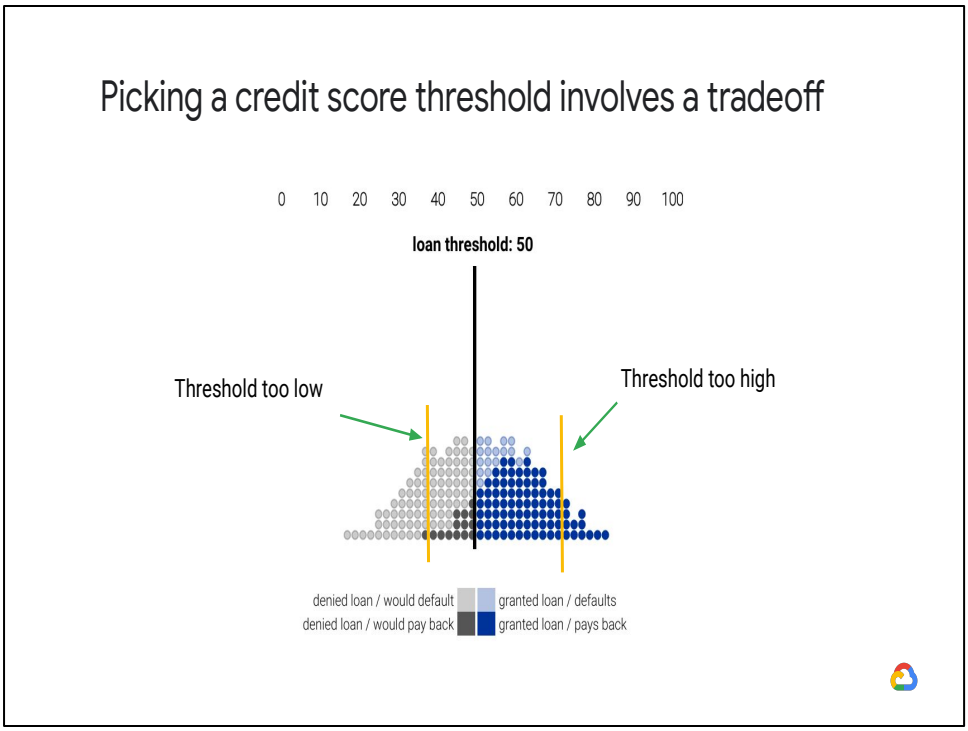


Let’s illustrate this approach using a toy classifier -- this is not a real model; it’s just a synthetic example to explain the concept.
The purpose of the model is to predict with high accuracy who will pay back their loan— the bank can use this model to help decide whether or not to give a loan to the applicant.

In the diagram that you see here,

- the dark dots represent people who would pay off a loan, and the light dots are those who wouldn't.
- the numbers at the top row represent credit score, which is simplified to a range of 0 to 100—where a higher score represents a higher likelihood of repaying the loan.

In an ideal world, you would work with statistics that cleanly separate categories, as you can see in the example on the left side. Unfortunately, it is far more common to see the situation at the right side, where the groups overlap.



Now, a single statistic like a credit score can stand in for many different variables, boiling them down to one number. You will see later in this specialization that most classification machine learning models return a probability and so the credit score here could stand in for that probability.

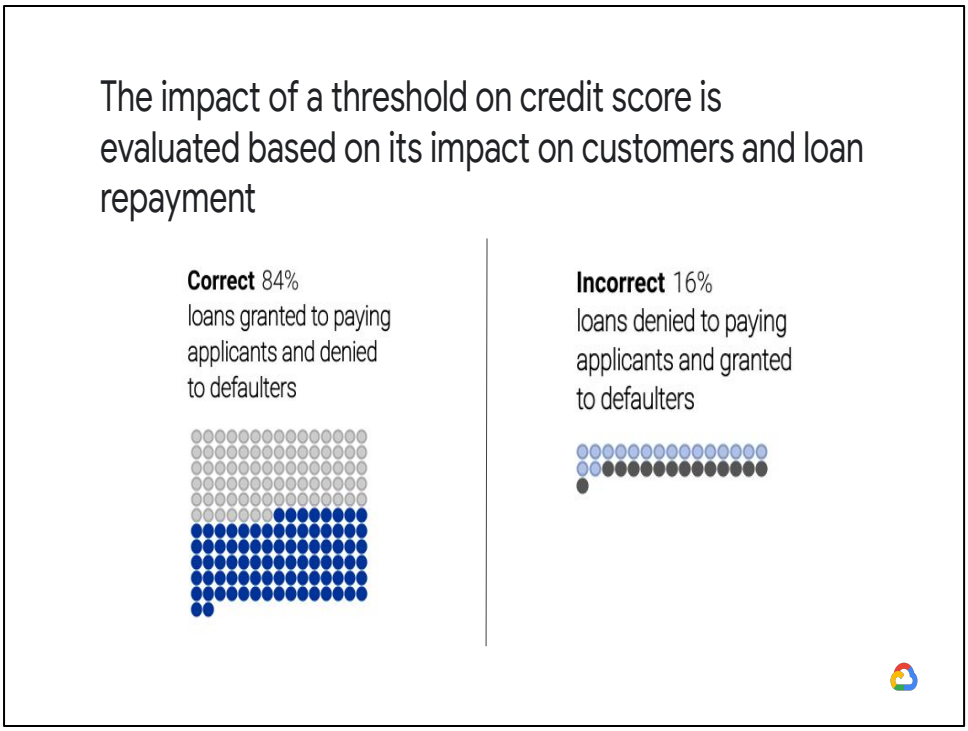
The resulting probability from the ML model, like the credit score, would factor in a lot of things, including income, promptness in paying debts, and so on—so the number might correctly represent the likelihood that a person will pay off a loan or default — **or it might not.**

This is where the idea of a setting a threshold comes in. You can pick a particular cut-off, or threshold, and people whose credit scores are below it are denied the loan, and people above it are granted the loan.

As you can see in this diagram, picking a threshold requires some tradeoffs. Too low, and you may give loans to many people who default. Too high, and many people who deserve a loan won't get one.

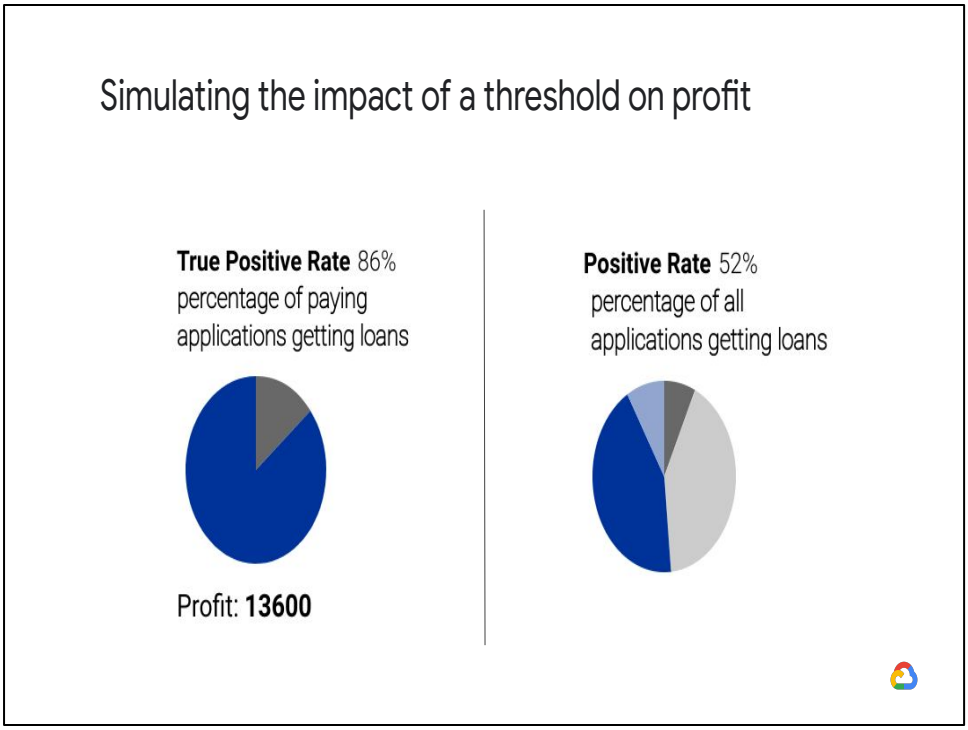
So what is the best threshold?

slight pause



What threshold to use depends on your goals and motivations. One goal might be to maximize the number of correct decisions, as you can see in this diagram, where:

- On the left, the dark blue dots represent loans that were granted and paid back and the light gray dots represent loans that were denied because they would default — all of these dots would represent correct predictions
- On the right, the light blue dots represent loans that were granted and defaulted and the dark gray dots represent loans that were denied to people that would have actually paid them off — these dots represent incorrect predictions

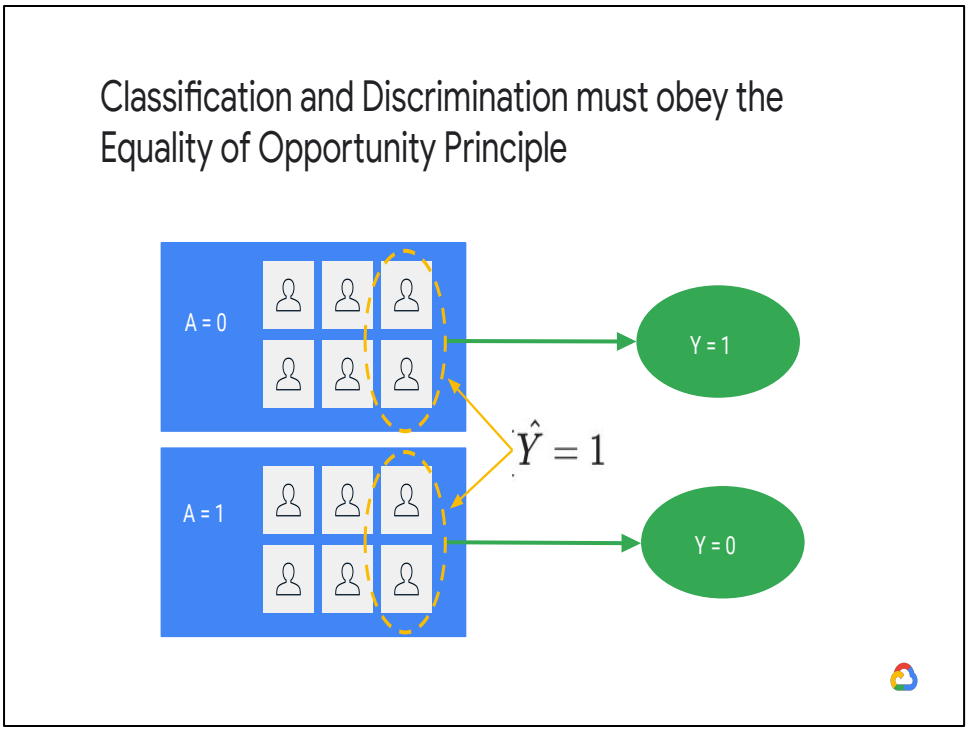


But some decisions are financially more costly than others. Perhaps there is a category of loans, perhaps loans for a 15-year mortgage, that are more profitable than other loans. So, you may not want to treat all decisions the same.

So, another goal, in a financial situation, might be to maximize (not the number of correct decisions) but the overall profit. At the bottom of the diagram is a readout of a hypothetical "profit" based on our estimate of the profit associated with each loan.

So the question then becomes: what is the most profitable threshold? And does it match the threshold with the most correct decisions?

Questions like these become particularly thorny when a statistic like a credit score ends up distributed differently between two groups.

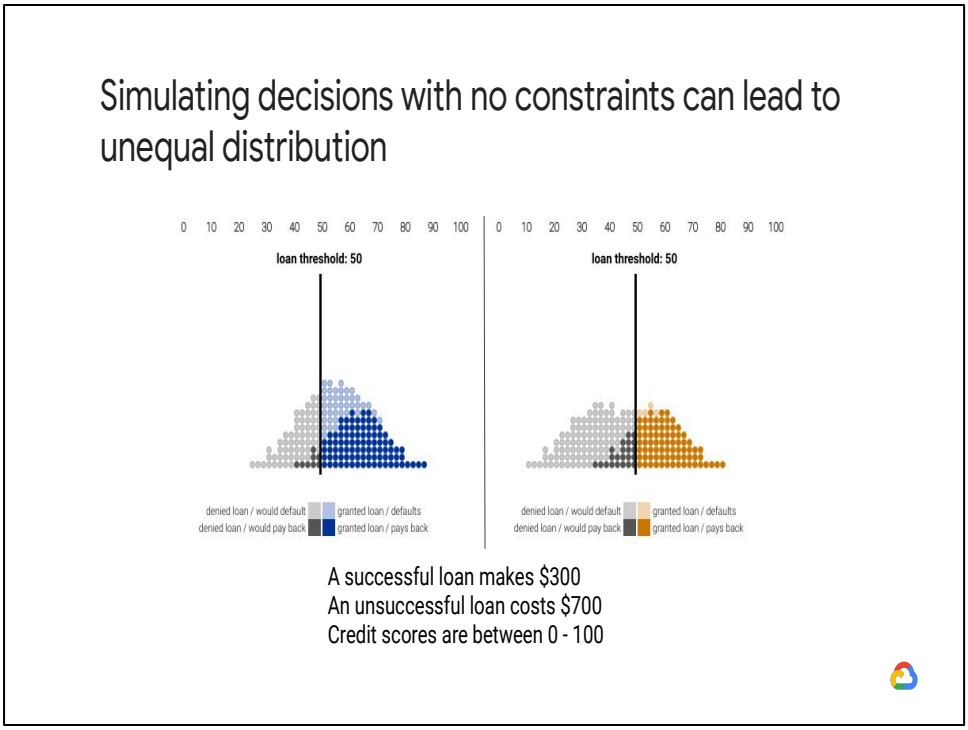


A thorny problem.
This is where equality of opportunity can come in. The formal setup for equality of opportunity looks something like this:

- Let's say A is some protected attribute. For simplicity, let's treat A as binary and represent membership of some protected group. Now, you're not a lawyer, so you can not tell what constitutes a protected group in your application area -- you should talk to your company's legal department to find out what is protected and what isn't. But to give you an idea, in the United States, federal law protects employees from discrimination based on age. So, depending on the application you are building, age might be a protected group.
- You have a binary outcome, which we'll call Y—and interpret Y=1 as the desirable outcome (in this case, acceptance for a loan). Consider Y in this example as your ground truth or label.
- But we're building a model of Y—so we'll call that Y^ (Y HAT), our predictor. In our example, the predictor is always a threshold predictor defined using a score between 0 and 1. The predictor may use thresholds that depend on A where we can use different thresholds for different groups.

So the idea here is that individuals in A who qualify for a positive outcome should have the same chance of getting positively classified as the individuals who are not in A.

More formally speaking, this desire coincides with an equal true positive rate in both groups—and this is the principle behind equality of opportunity.

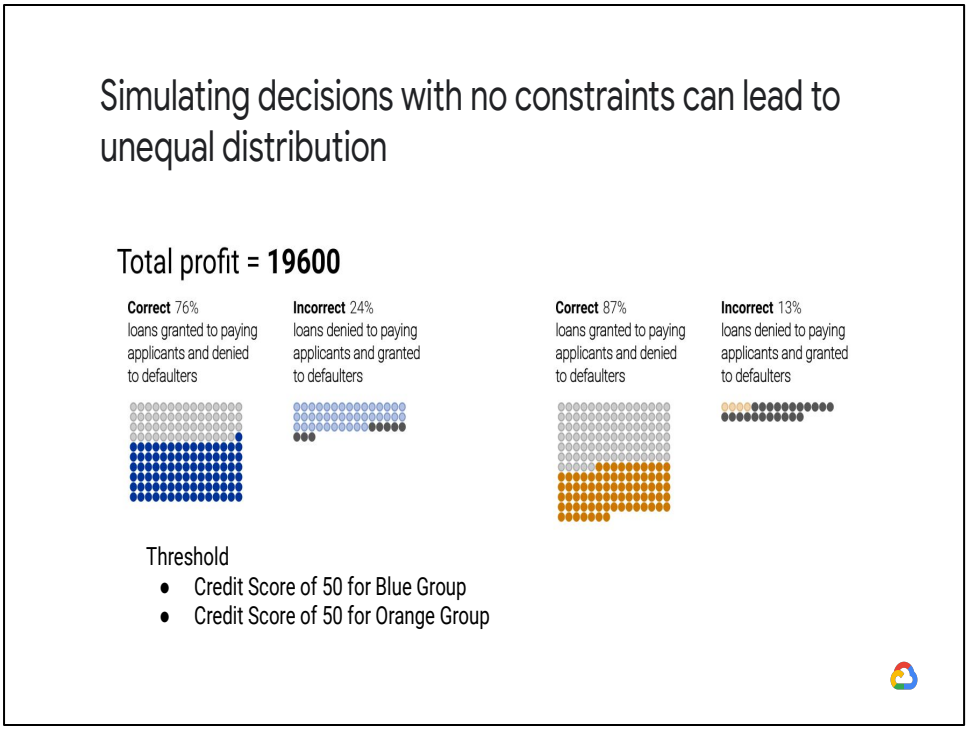


So now that you've formally defined what the principle behind equality of opportunity is, let's walk through the loan predictor example once more.

In this scenario, you have two groups of people, "blue" and "orange." And let's say you're interested in making small loans, subject to the following conditions:

- A successful loan makes $300
- An unsuccessful loan costs $700
- Everyone has a credit score between 0 and 100

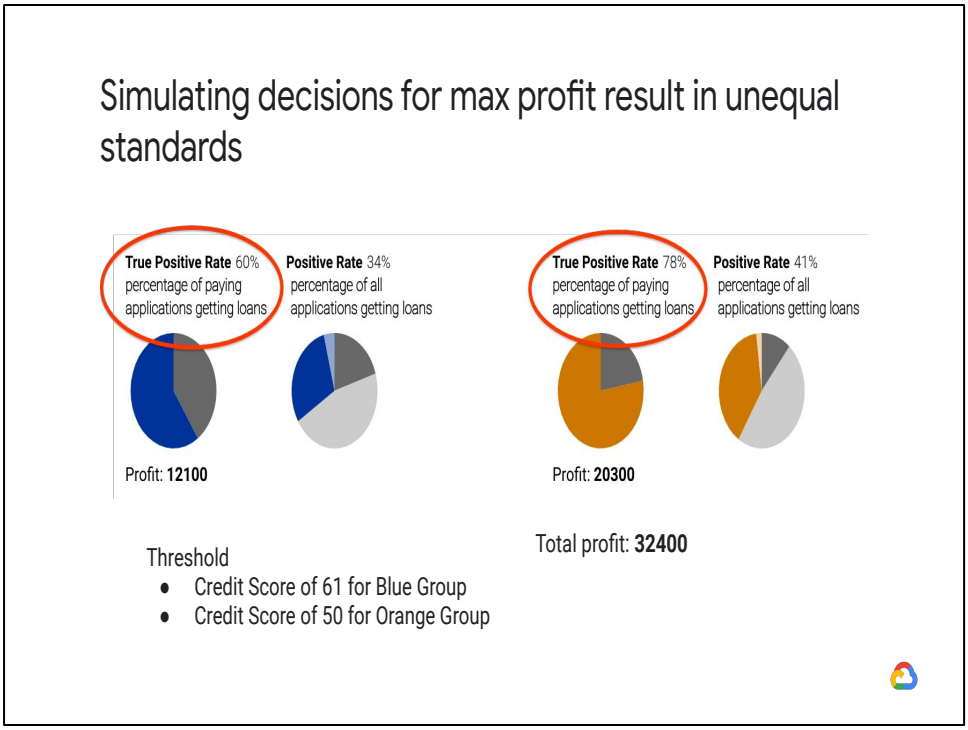
And let's first start by setting the threshold at a credit score of 50



Now because the distributions of the two groups are slightly different, setting the threshold to a credit score 50 gives you some decent results

For the blue folks, a threshold of 50 leads to correct decisions 76% of the time
For the orange folks, a threshold of 50 leads to correct decisions 87% of the time.

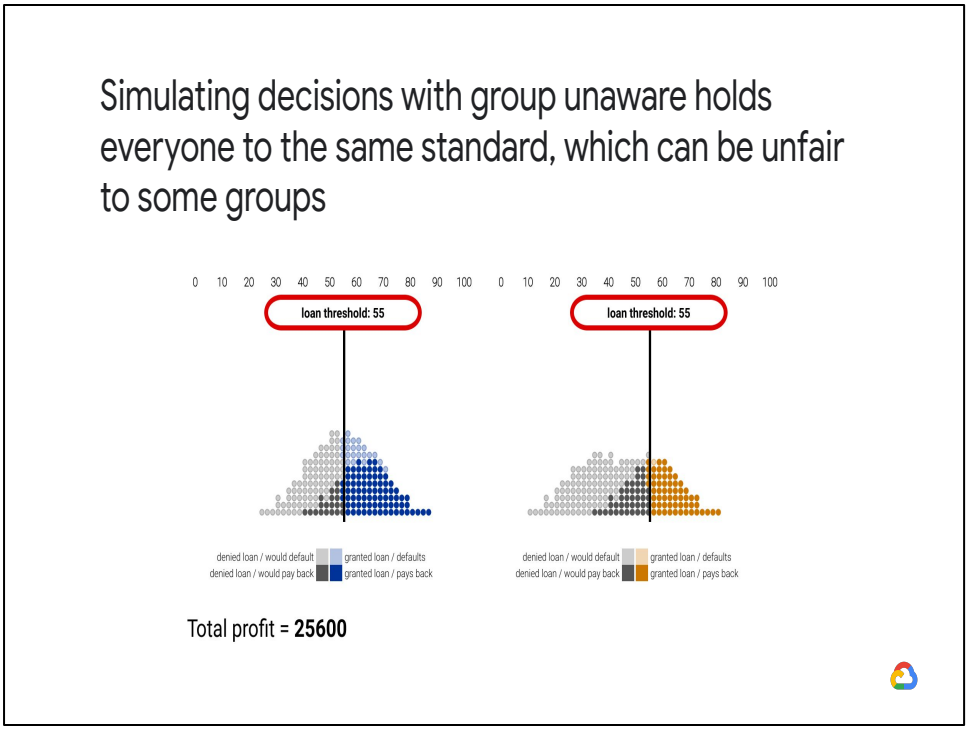
So this default threshold suggests, is, it's better to be in the orange group than in the blue group.
So, there's room for improvement.



Now let's say you set your thresholds to maximize profit.

If you look for pair of thresholds that maximize total profit maybe you'll see that the blue group is held to a higher standard than the orange one. And that's depicted by the increase in dark gray shaded region (which are those that were denied a loan even though they would have paid it back)

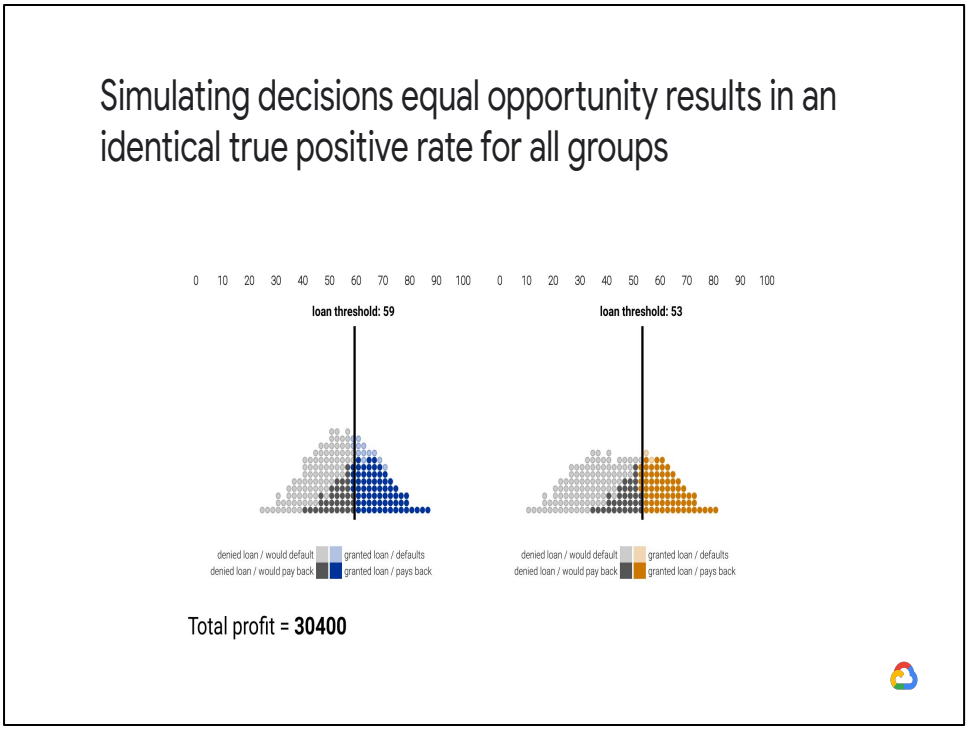
That could be a problem—and one obvious solution for the bank is to not just pick thresholds to make as much money as possible.



Another approach would be to implement a group-unaware approach, which holds all groups to the same standard.
So, you'll use the same threshold (55) for all groups.
Is this really the right solution, though?

For one thing, if there are real differences between two groups, it might not be fair to ignore them—for example, women generally pay less for life insurance than men, since they tend to live longer.

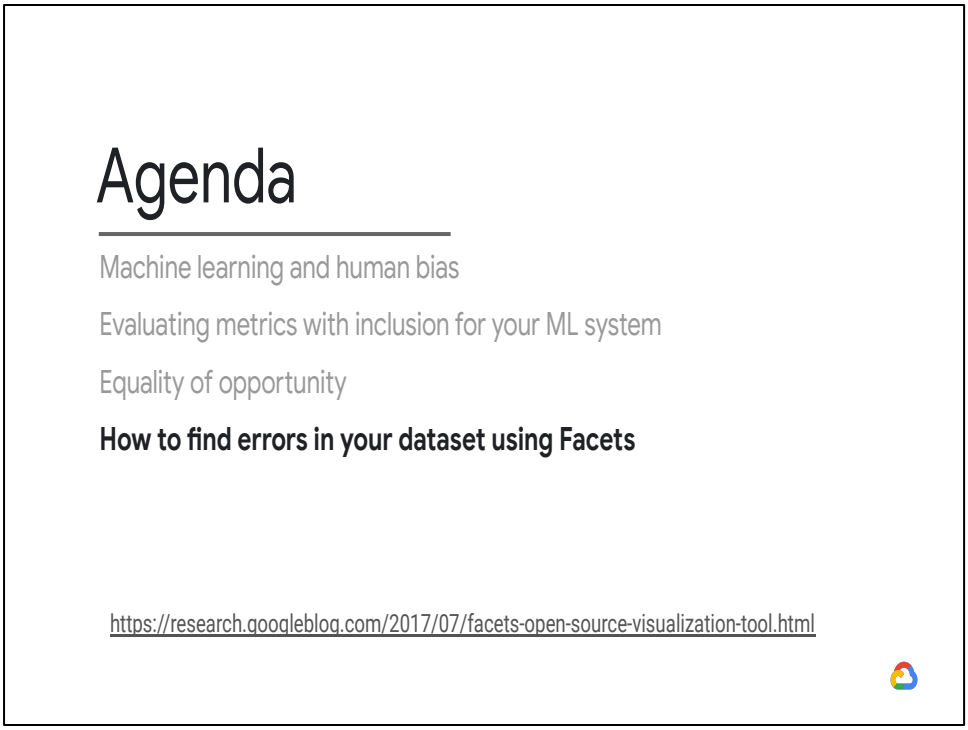
But there are other, mathematical problems with a group-unaware approach even if both groups are equally loan-worthy. In the example above, the differences in score distributions means that the orange group actually gets fewer loans when the bank looks for the most profitable group-unaware threshold.



But if you were to take the equality of opportunity approach, then in this example, among people who would pay back a loan, blue and orange groups do equally well. This choice is almost as profitable as optimization for maximum profits, and about as many people get loans overall.

Here, the constraint is that of the people who can pay back a loan, the same fraction in each group should actually be granted a loan. Or, using some of the jargon I described in the earlier section, the "true positive rate" is identical between groups.

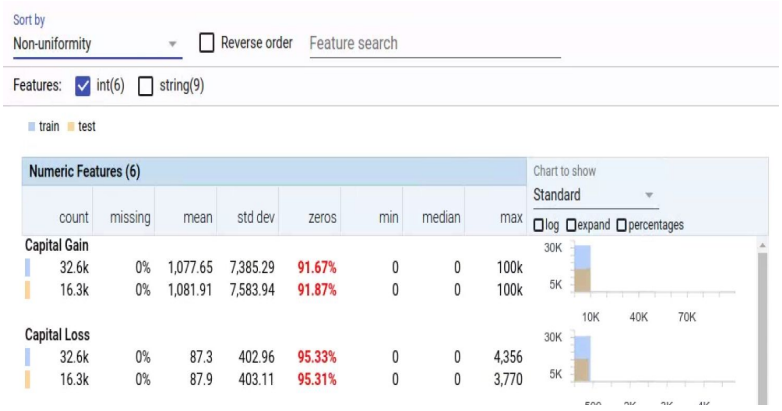
So the takeaway to all of this is that it's possible to efficiently find thresholds that meet any of these criteria. When you have control over your machine learning system, using these definitions can help clarify core issues. If your model isn't as effective for some groups as others, it can cause problems for the groups that have the most uncertainty. Restricting to equal opportunity thresholds transfers the "burden of uncertainty" away from these groups and onto you, the creator of the model. Doing so provides with the incentive to invest in better classifiers.



So you've covered some of the ways in which you can make your machine learning model more inclusive through evaluation metrics—but getting the best results out of a model requires that you truly understand your data. The challenge here though is that sometimes datasets can contain hundreds of millions of data points, each consisting of hundreds (or even thousands) of features, making it nearly impossible to understand an entire dataset in an intuitive fashion.

The key here is to utilize visualizations that help unlock nuances and insights in large datasets—and in this section will talk about an open-source data visualization tool called Facets. Facets was developed at Google, and we open-sourced it in July 2017 -- Facets can help you make machine learning more inclusive.

Facets gives users a quick understanding of the distribution of values across features

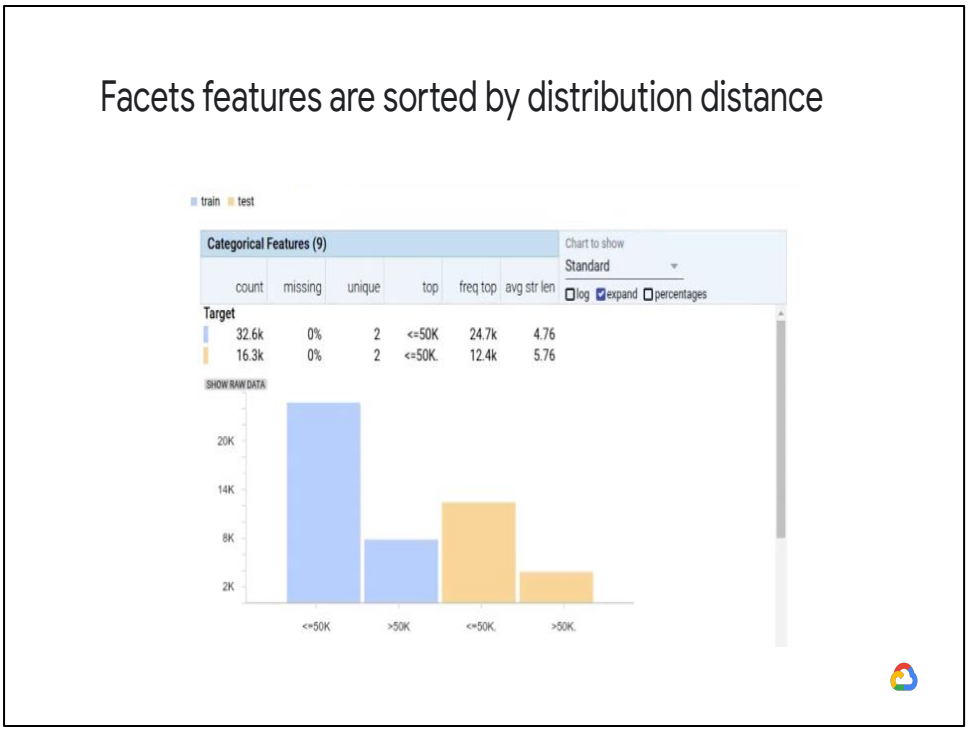


So there's two parts to Facets: Overview and Dive. In this slide, you're seeing a screenshot of Facets Overview, which automatically gives you a quick understanding of the distribution of values across the features of their datasets.

The example you're seeing in this slide comes from the UCI Census data. This data was extracted from the 1994 Census bureau database, which contains some anonymized information about the US population. The information in this data set includes demographic and employment-related variables, such as age and salary. This data set was put together by the research community and is often set up as a prediction task to determine whether a person makes over $50K USD a year.

Multiple datasets, such as a training set and a test set, can be compared on the same visualization. Common data issues that can hamper machine learning are pushed to the forefront, such as: unexpected feature values, features with high percentages of missing values, features with unbalanced distributions, and feature distribution skew between datasets.

What you're seeing here are two of the numeric features of the UCI Census datasets: capital gain and capital loss. The features are sorted by non-uniformity, with the feature with the most non-uniform distribution at the top. Numbers in red indicate possible trouble spots, in this case numeric features with a high percentage of values set to 0. The histograms at right allow you to compare the distributions between the training data (blue) and test data (orange).



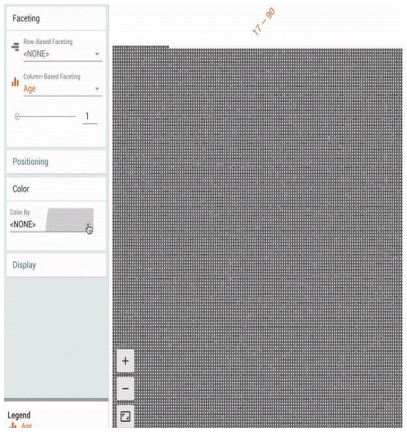
Facets Overview can also visualize categorical features. In this example, what you're seeing here is a breakdown of the "Target" feature, which is the label that represents whether or not the person earned an annual salary of more than $50K. But in particular, you're looking at all the instances where the annual salary is less than or equal to $50K.

Do you notice anything suspicious about the "Target" feature?

Notice that the label values differ between the training and test datasets, due to a trailing period in the test set ("<=50K" vs "<=50K.") Facets Overview even went so far as to sort these discrepancies by distribution distance, with the feature with the biggest skew between the training (blue) and test (orange) datasets at the top. This can be seen in the chart for the feature and also in the entries in the "top" column of the table.

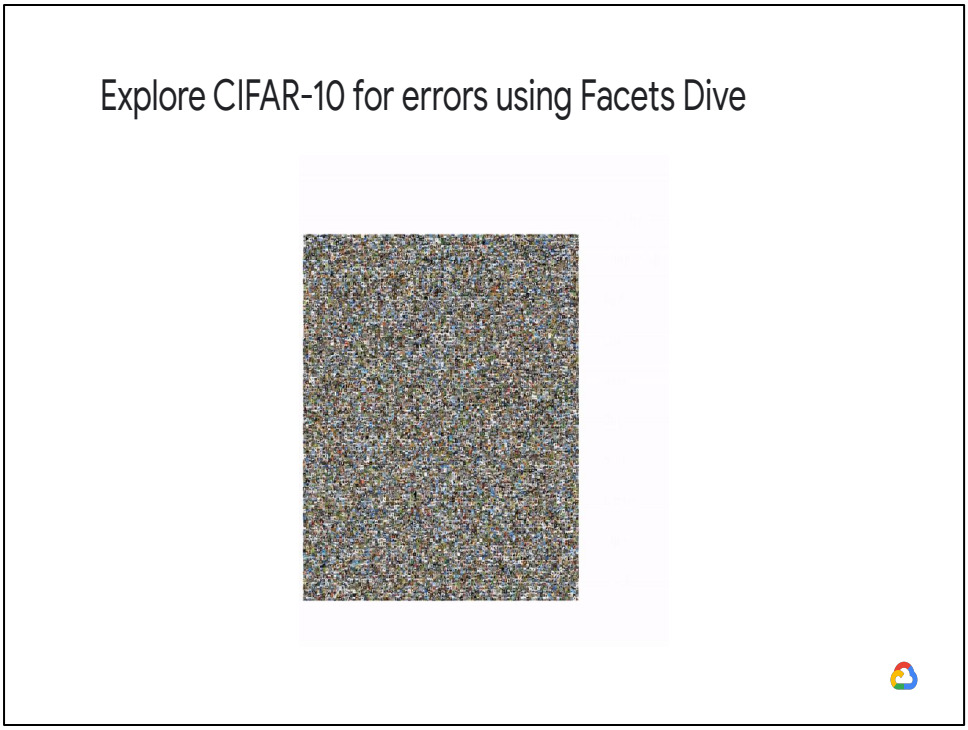
Encountering a label mismatch like this would cause a model trained and tested on this data to not be evaluated correctly.

Facets Dive provides an easy-to-customize, intuitive interface



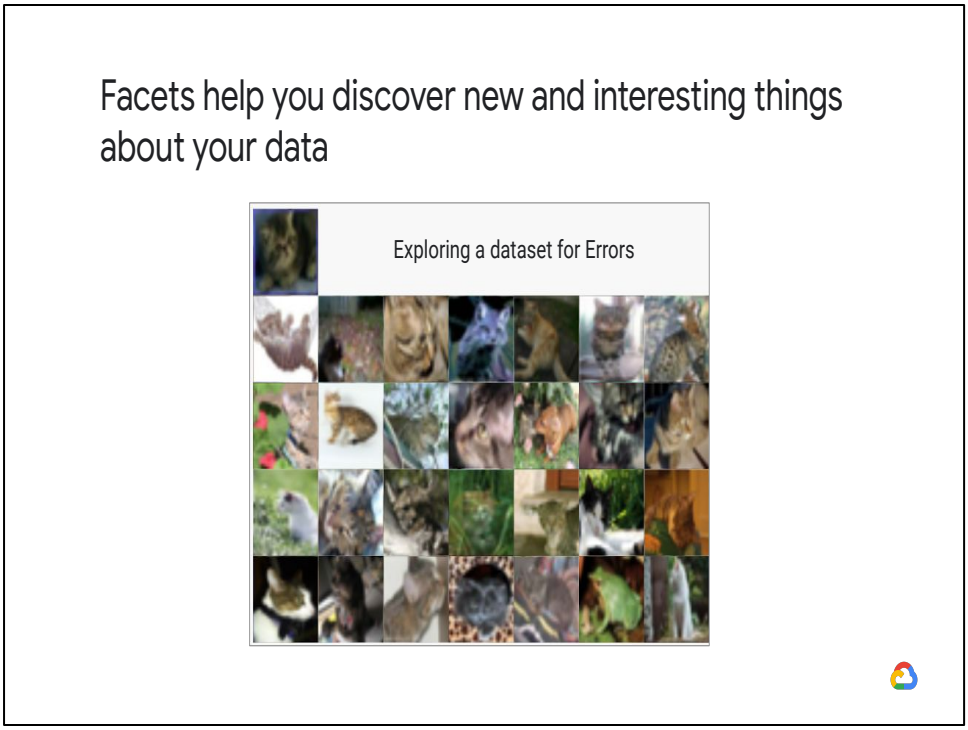
Now, shifting over to Facets Dive, you can see in this animated slide that it provides an easy-to-customize, intuitive interface for exploring the relationship between the data points across the different features of a dataset. With Facets Dive, you control the position, color and visual representation of each data point based on its feature values.

More specifically, in this example, Facets Dive is displaying all 16281 data points in the UCI Census test dataset. The animation shows a user coloring the data points by one feature (“Relationship”), faceting in one dimension by a continuous feature (“Age”) and then faceting in another dimension by a discrete feature (“Marital Status”).



In Facets Dive, if the data points have images associated with them, the images can be used as the visual representations—so it's not only limited to categorical or numerical features.

The example you see here in this slide comes from a research-based image data set that contains many objects and animals in the world used to train an image classifier. The ground truth labels are arranged by row and the predicted labels are arranged by column. This configuration produces a confusion matrix view, allowing us to drill into particular kinds of misclassifications.



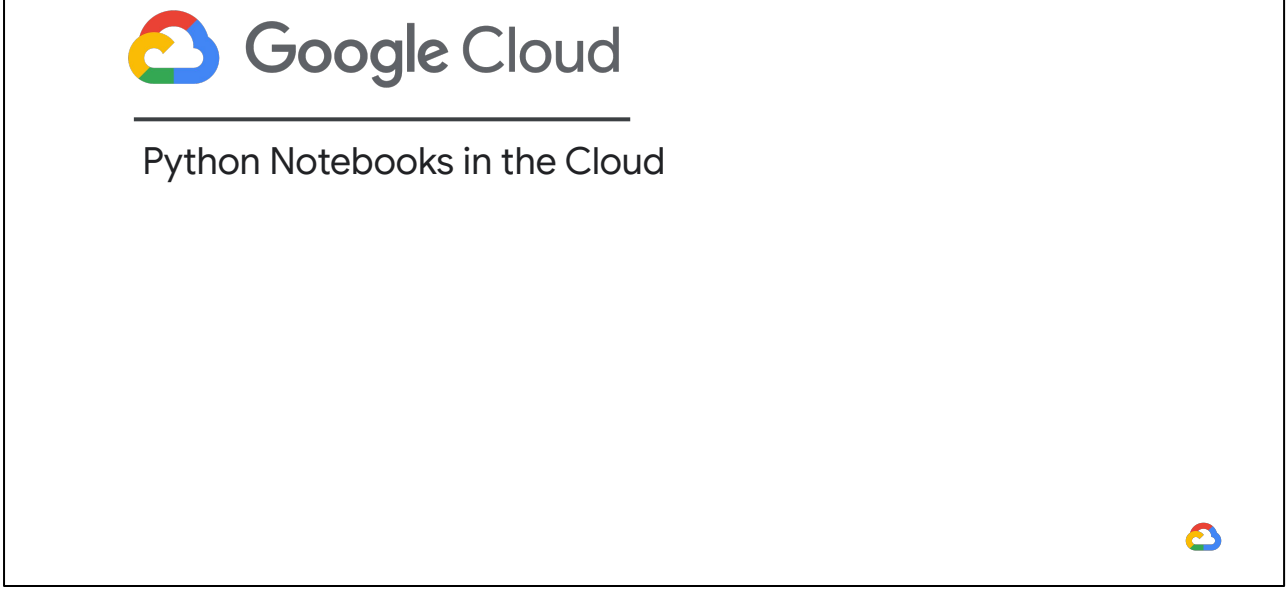
In this particular example, the ML model incorrectly labels some small percentage of true cats as frogs.

Can you spot the frog-cat in this image?

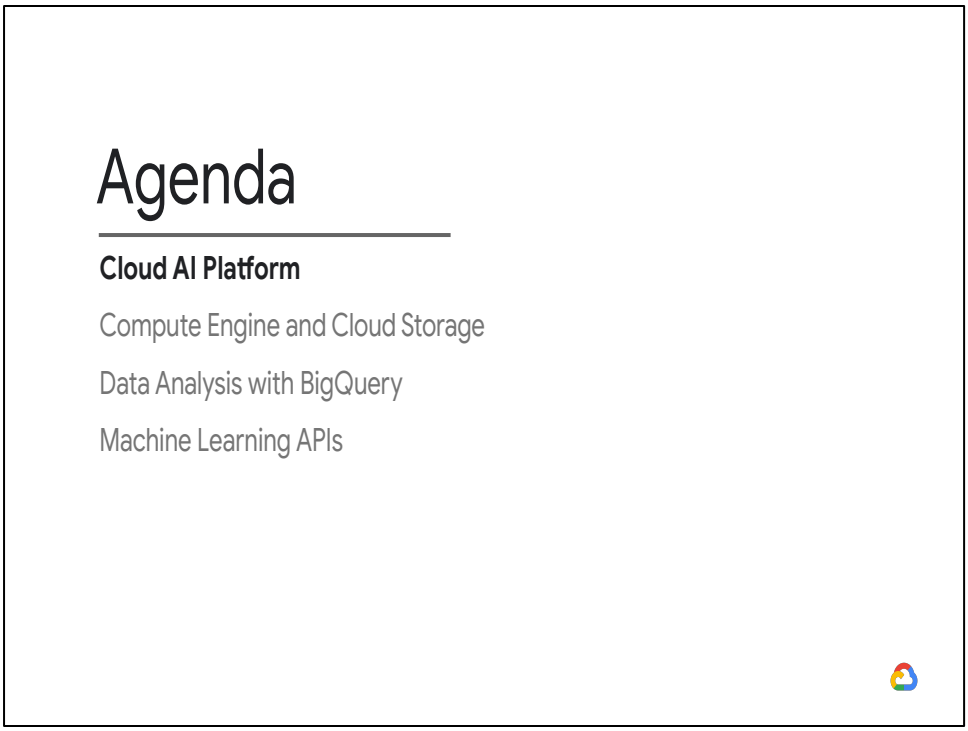
The interesting thing you find by putting the real images in the confusion matrix using Facets Dive is that one of these "true cats" that the model predicted to be a frog is actually a frog from visual inspection.

With Facets Dive, we can determine that this one misclassification wasn't a true misclassification of the model, but instead incorrectly labeled data in the dataset.

So the hope here is that tools such as Facets help you discover new and interesting things about your data that lead you to create more powerful, accurate and inclusive machine learning models.



Welcome to ‘Python notebooks in the cloud’.



Cloud AI Platform is basically the Integrated Development Environment you'll be using to write your code in this course.

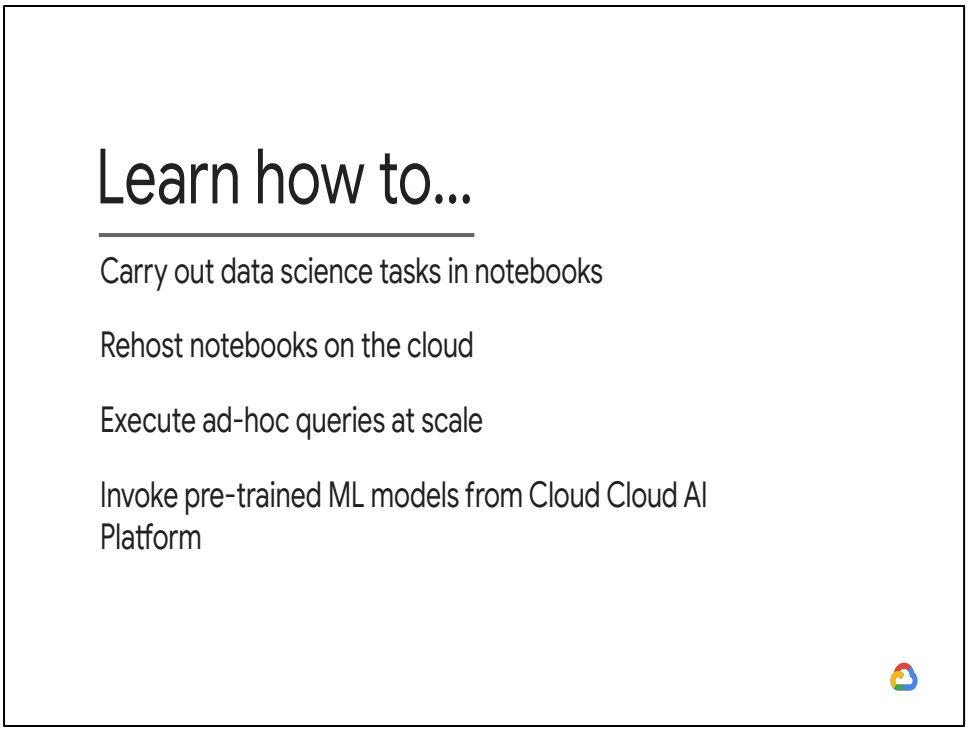
Cloud Platform notebooks run on virtual machines. Because of that, we'll talk about Compute Engine and Cloud Storage. Why?
Two things follow from the fact that Cloud Cloud AI Platform runs on a VM.
First, it means that you can actually control and change what sort of machine is running your notebook by—for example—giving it more memory or adding a GPU, *without having to rewrite your notebook from scratch. Rehosting a notebook on a more powerful machine is trivially easy.*

Second, VMs are ephemeral. Consequently, anything that you want to persist, you must store outside of the VM. The best place to do that, especially for large binary files, is in Cloud Storage. So, after reviewing how Compute Engine works, we'll review the basics of Cloud Storage. The notebooks, themselves, we will store in a cloud repository so that they are under revision control.

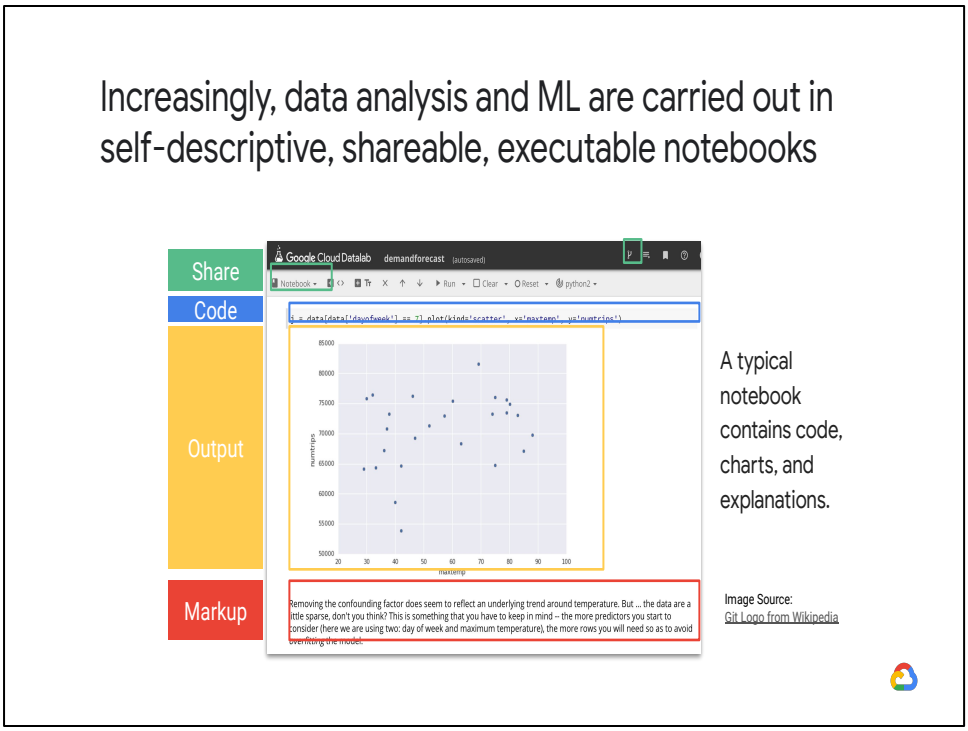
Finally, you'll do a hands-on lab so you can get hands-on with Cloud AI Platform Notebooks. You'll seeyou how to use Cloud AI Platform Notebooks together with BigQuery, which is a managed data analysis service on the cloud that will allow you to execute ad-hoc queries at scales and speeds that are not possible with traditional database systems.

Then, you'll look at how to invoke pre-trained ML models, and do this from within

Cloud AI Platform Notebooks.



We will learn how to develop machine learning models in Python notebooks, where the notebook server is on the cloud and execute ad-hoc queries using serverless technologies. Not every machine learning model needs to be built from scratch -- also in this module, my colleague Sara will show you how to invoke pre-trained machine learning models.



Do you use IPython or Jupyter notebooks today? Increasingly, data scientists work in self-descriptive, shareable, executable notebooks whenever they want to do data analysis or machine learning.

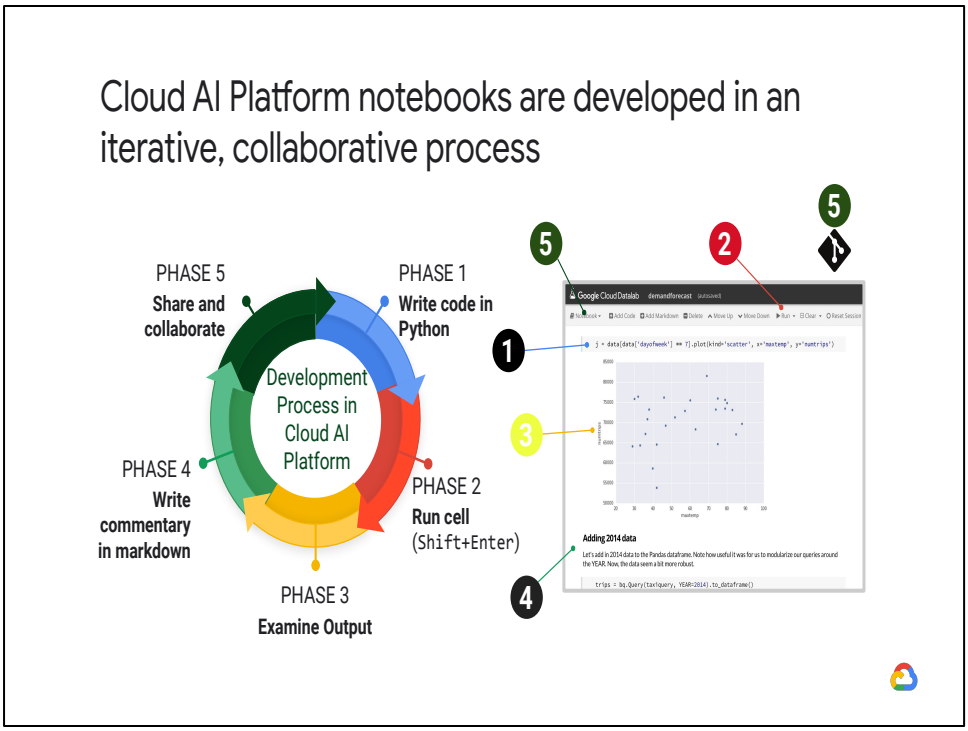
Cloud AI Platform is based on Jupyter and it's open source.

This is what the interface to Cloud AI Platform looks like. Notice how there are code sections interleaved with markup and output. This interleaving is what makes this style of computing so useful.
Data analysis and machine learning are commonly carried out in notebooks like this.

You can execute the code by either clicking the Run button or by pressing Shift + Enter.
Notice that output here isn't just command line output; it's charts as well.
The red section contains markup, so you can explain why you're doing what you're doing.
One of the green sections contains a button for exporting the notebook as a standalone file. The other green section is what you'd click if you wanted to commit your changes to a code repository in GCP. In the lab, we'll show you how to pull from and commit to code repositories.

You can execute the code by either clicking the Run button or by pressing Shift + Enter.

Notice that output here isn't just command line output; it's charts as well.
The red section contains markup, so you can explain why you're doing what you're doing.
One of the green sections contains a button for exporting the notebook as a standalone file. The other green section is what you'd click if you wanted to commit your changes to a code repository, in GCP. In the lab, we'll show you how to pull from and commit to code repositories.



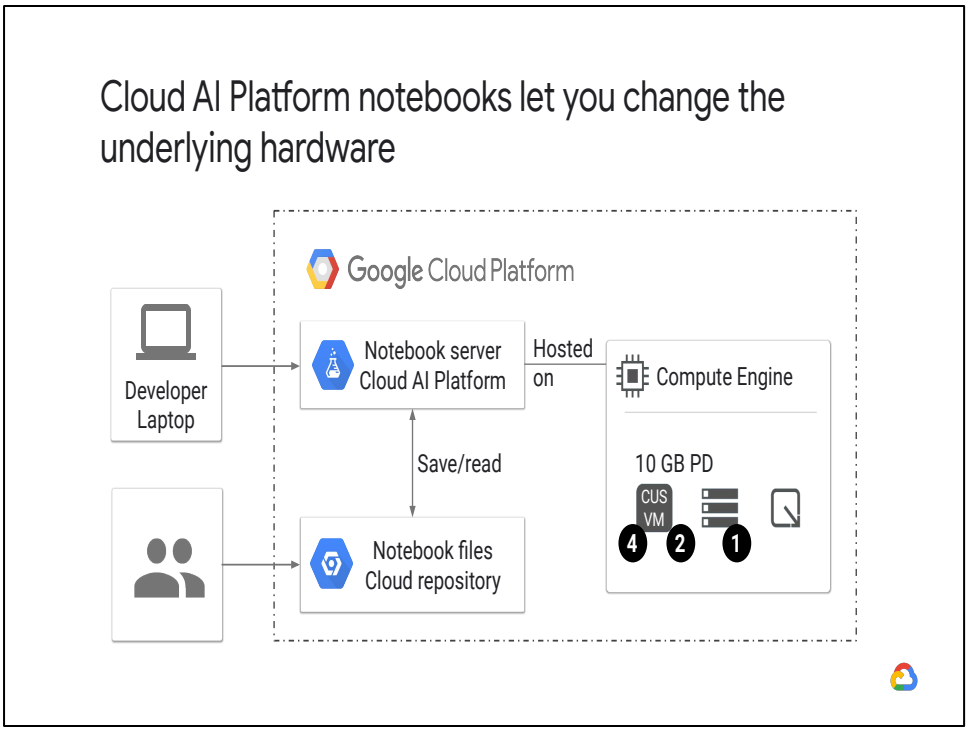
Have you used Google Docs? How is it different from documents edited in a desktop editor?
Have you filed taxes online? How is the experience different from doing your taxes in a desktop program?
There are lots of benefits, but one key aspect is collaboration. You don't have to email documents back and forth.
Imagine, you start doing scientific research, collaborating on a single result would be painful. You'd write some code and create a graph. Then, you would snapshot, create an image file, put into a doc, create a PDF, and send it to your collaborator.
A few hours later, your colleague would say, "Looks great, but could you add one more year's data? It looks kinda sparse."
And you'd go through the process all over again. Why? Because the PDF you'd sent was not editable. Round-trips took a long time.

Enter Python notebooks.You'd write the code, create the graph, write some commentary, and send the notebook link to your colleague. This way, when your colleague wants to add one more year of data, they would simply edit the cell, look at the new graph, and say, "See, it looks a lot better." And that's great; you now have a better notebook for the next step.

One problem with traditional notebooks: Who runs the server that hosts these pages? Whose machine? If it's yours, and your machine goes to sleep, then your colleague can't work!

When your cloud AI Platform notebook is hosted in the cloud, you can develop together quite easily.
And just as Google Docs are available even when your computer isn't on, so too are Cloud AI Platform notebooks, when you run them in the Cloud.

To share a notebook *within* a project, other users can simply "Cloud AI Platform connect" to the VM and work using the URL. Another way to share notebooks is through revision control systems (for example, git).

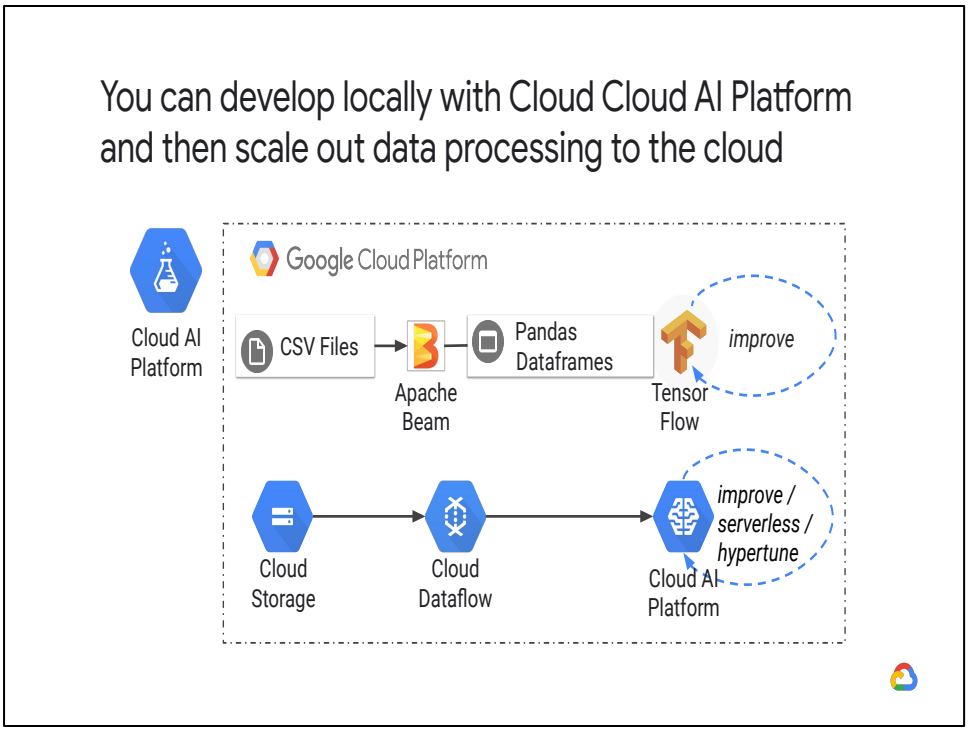


Cloud also frees from you being constrained by hardware limitations.

Not only can you run Cloud AI Platform on any Compute Engine machine you want, you can also change the machine specs after it's been provisioned.

You can go to the web console, find the running VM, stop it, and restart it with a new machine configuration.

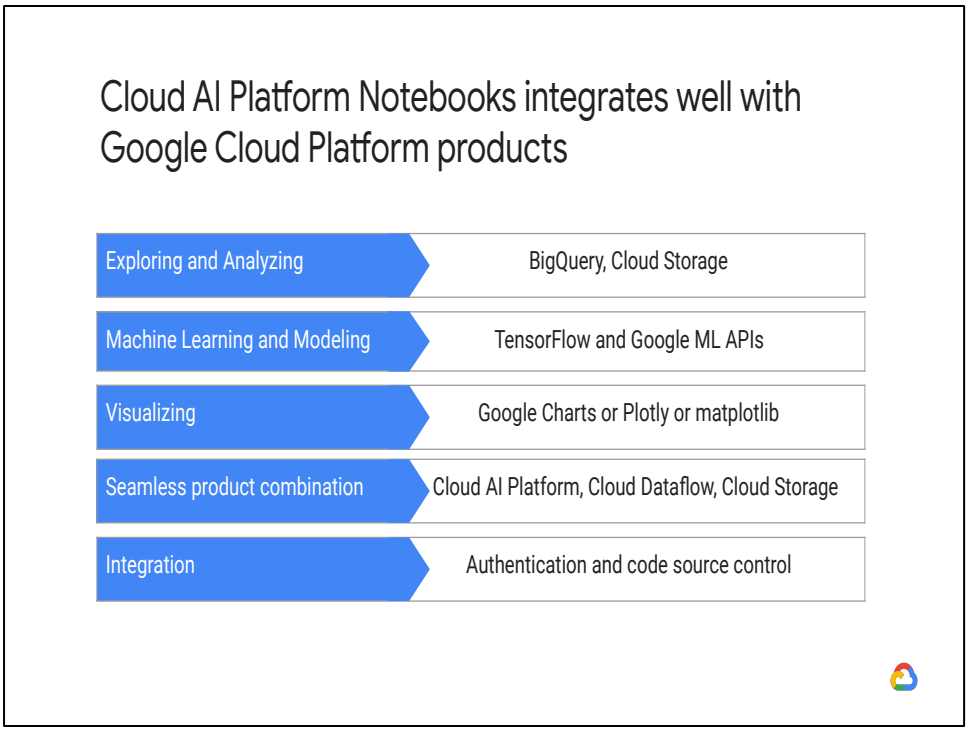
As a developer, to work in Cloud AI Platform, you simply connect to the VM that's running the notebook server.
The notebooks themselves can be persisted in git, so you can delete the VM if you don't need it anymore.
When you run the notebook, the computation is carried out on the VM.



Cloud AI Platform works with the same technologies that you're comfortable with, so you can start developing now, and then work on scale later.

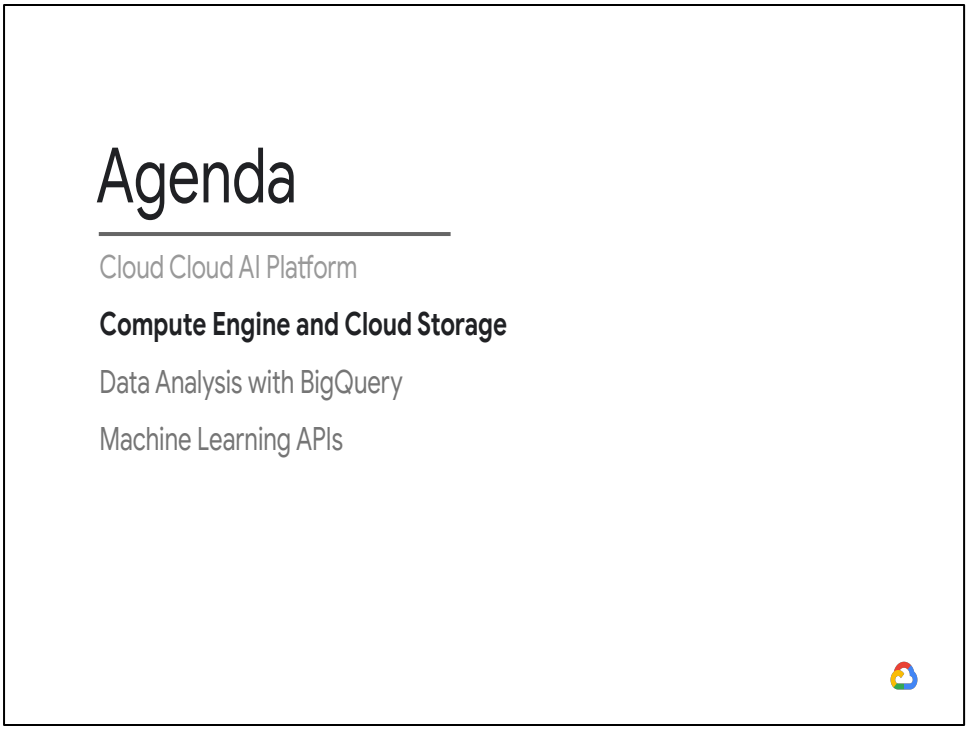
For example, you'll be doing an exercise where you read from a .csv file. You could then process in Pandas and Apache Beam before training a model in TensorFlow, and then improve the model through training.

Eventually though, when you are ready to scale, you can use Google Cloud Storage to hold your data initially, process it in Cloud Dataflow on an ephemeral cluster, and then run distributed training and hyper-parameter optimization in Cloud ML Engine.



And you can do this because Cloud AI Platform integrates seamlessly with all other GCP projects. In a few minutes, you'll do a lab that shows you how easy it is to connect to BigQuery and harness thousands of machines to explore and analyze your data. You can also write TensorFlow code and connect with Google ML APIs; authentication is a breeze. You can even start big computational jobs in CLME and Dataflow.

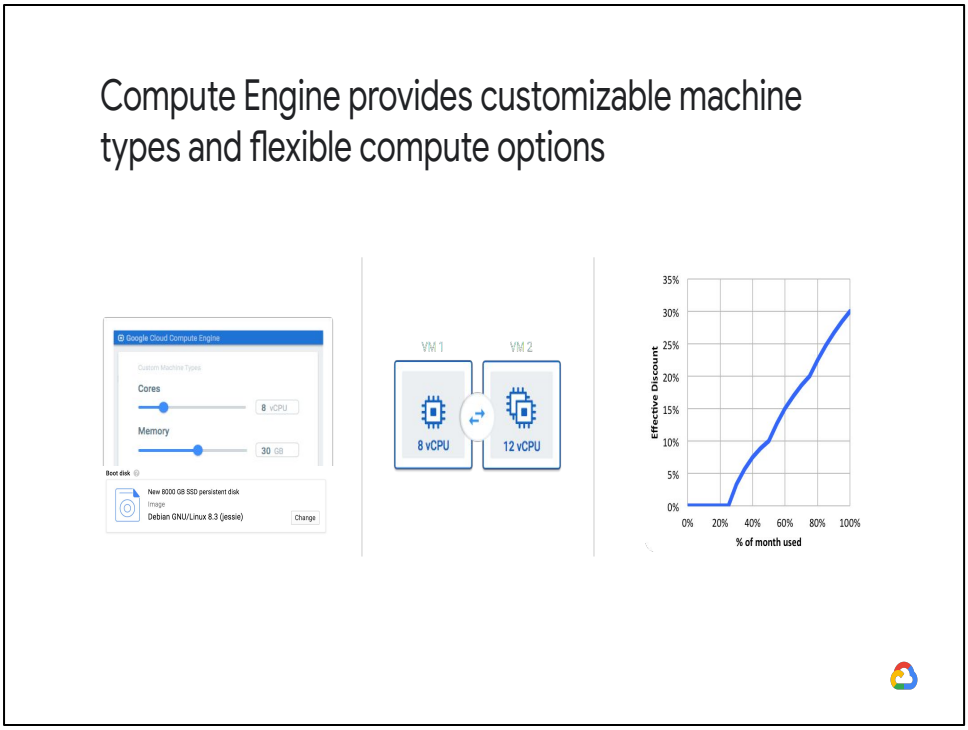
Of course, you can do all the things you can do in a Python notebook—doing analysis with Pandas or visualizing query results using Plotly or seaborn.



So, let's talk about Compute Engine and Cloud Storage. It is useful to know how compute instances on cloud work, because the Cloud AI Platform instance is going to run on these. For persistent data in the cloud, you will use Cloud Storage. So, you need to understand Cloud Storage as well.

Think of Compute Engine as a global, distributed CPU, and Cloud Storage as a global, distributed disk.
Cloud AI Platform, though, is a single-node program, so it runs on a single Compute Engine instance. However, when you launch off Dataflow jobs or Cloud ML Engine jobs, you can kick off the processing to many Compute Engine instances.

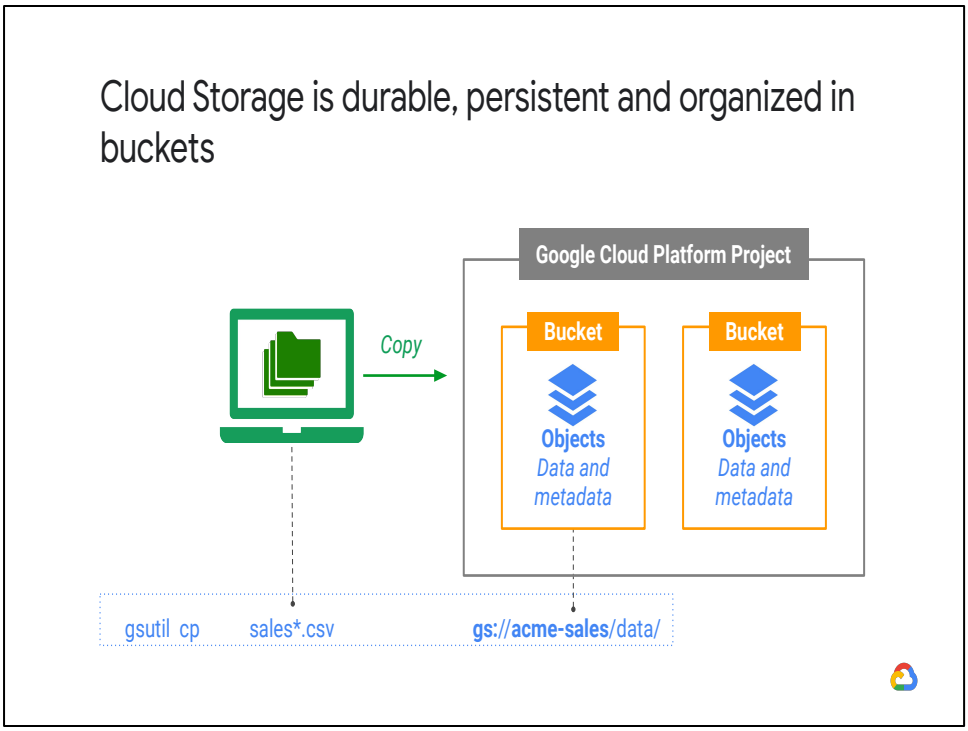
https://pixabay.com/en/hard-drive-hdd-technology-digital-870699/ (cc0)



Compute Engine essentially allows you to rent a virtual machine on the cloud to run your workloads.
What are some of the things you can customize? cores, memory, disk size, operating system.
Load balancing, networking, and so on all come “baked in.” Then enjoy pricing simplicity and agility.

You are not tied into your initial choices; you can always change them.
And the billing discounts are automatic based on how much you use a machine.

https://cloud.google.com/custom-machine-types/
https://cloud.google.com/compute/pricing



Disks attached to Compute Engine instances are fast, but ephemeral. When the VM goes away, the disk goes away.
Well, Google also offers persistent disks, but let's ignore that for now.

Cloud Storage is durable (that is, blobs in Cloud Storage are replicated and stored in multiple places)
Cloud Storage is also accessible from any machine. Because of the speed of the network (petabit bisectional bandwidth, which means 100,000 machines can talk to each other at 10 GB/s), you can directly read off Cloud Storage. In fact, this is what you will do from our TensorFlow programs.

The purpose of Cloud Storage is to give you a durable, global, file system. How is it organized?
A typical Cloud Storage URL might look like gs://acme-sales/data/sales003.csv

Acme-sales is called a bucket. The name is globally unique. Think of it like a domain name in an internet URL. The way to get a globally unique bucket name is to use a reverse domain name (in which case, Google Cloud Platform will ask you to prove that you own the domain name in question) or simply use your project id. Unless you are extremely unlucky, your project id (which is also globally unique) will not have been already used for a bucket name.

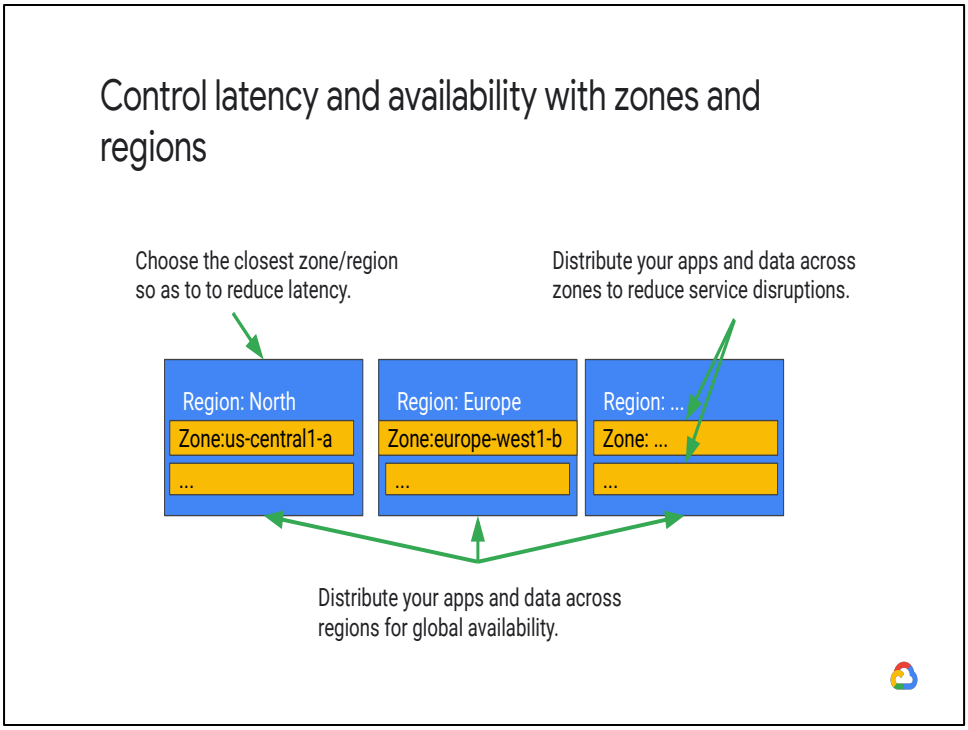
The rest of the gs:// URL is, by convention, like a folder structure, with the complete gs:// URL referring to an object in Cloud Storage.

How do you work with it? You can use gsutil. This is a command-line tool that comes with the Google Cloud SDK.

If you spin up a Compute Engine instance, gsutil is already available. On your laptop, you can download the Google Cloud SDK to get gsutil. gsutil uses a familiar UNIX command syntax. mb, rb are make-bucket and remove-bucket, for example. Instead of command-line, you can also use the GCP Console, a programming language, or REST API

Here, you are copying (cp) a bunch of files to the the Cloud Storage location.

https://cloud.google.com/storage/docs/overview



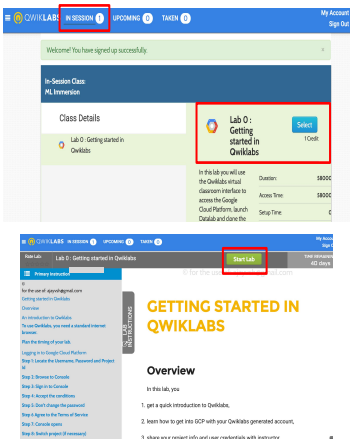
Remember, Cloud Storage objects are durable. This means that they are stored redundantly. You also get edge caching and failover simply by putting your object in Cloud Storage.

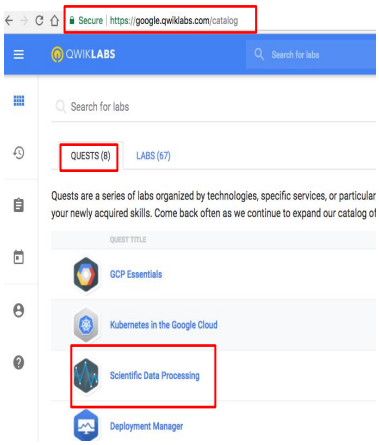
However, just because Cloud Storage is a global file system doesn't mean that you can forget about latency considerations. You are better off storing the data close to your compute nodes. However, you should also distribute your apps and data across multiple zones to protect yourself against service disruptions; for example, if a zone suffers a power outage. Leverage zones in different regions for additional redundancy.
A zone is an isolated location within a region. It's named <region-name>-<zone-letter>.

Distribute your apps and data across regions for global availability.

https://cloud.google.com/about/datacenters/

Use Qwiklabs to try out Google Cloud





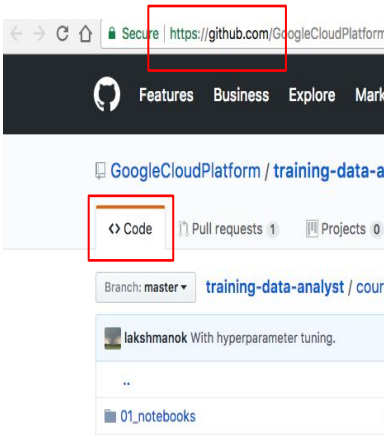
Our goal with Qwiklabs in this course is to provide a convenient way to try out Google Cloud in an environment that is actually Google Cloud, but doesn't result in your getting billed over and above what you paid for the Coursera course. So, you will get a temporary GCP account for each lab. It will be timed, and then the lab will go away.

You can come back and take more Qwiklabs at any time, to practice what you have learned. Particularly useful are the the Quests. There are more of them created all the time.

Source code for labs is on GitHub

https://github.com/GoogleCloudPlatform/training-data-analyst/tree/master/courses/machine_learning/deepdive

Later, practice taking the lab apart and trying to build it yourself on your own GCP account (strongly recommended).



The source code for all the labs is on GitHub. It's open source. You can always refer to this code and use it as a starting point for your code. You don't need to log in to Qwiklabs for that. You can, of course, also do these labs in your own GCP account. In fact, it is suggested that a great way to learn is to try to do the lab yourself, using the GitHub code as a starting point. You will learn a lot that way.

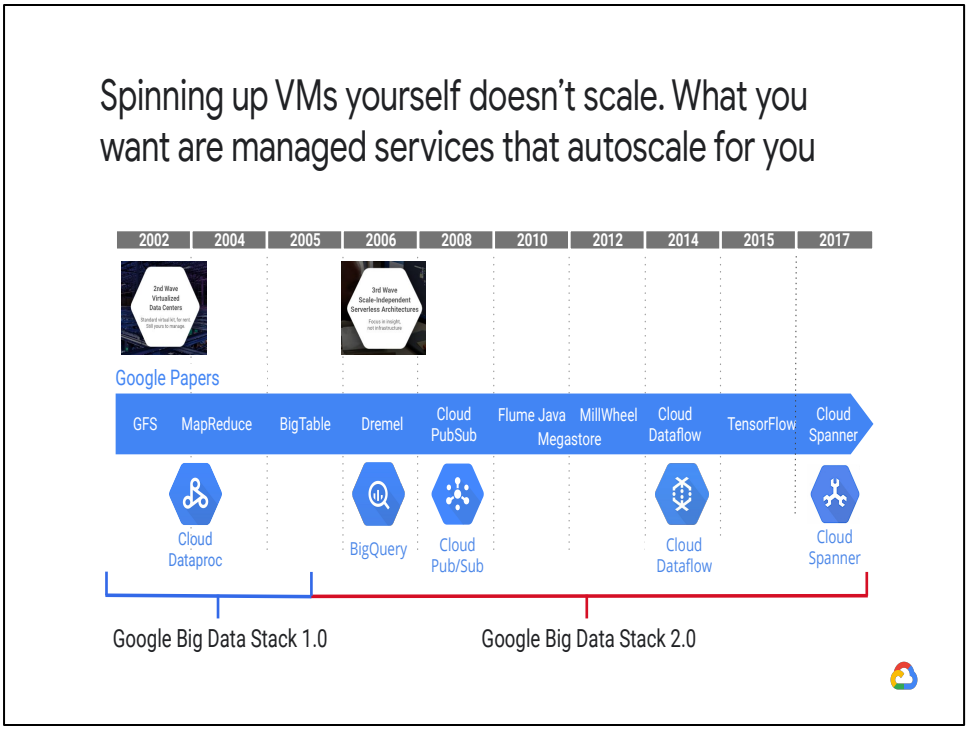
Agenda

Cloud Cloud AI Platform

Compute Engine and Cloud Storage

Data Analysis with BigQuery

Machine Learning APIs



Remember that Cloud AI Platform is a way to try things locally, but then scale it out to the cloud using managed services.

At Google, you may have invented MapReduce, but by 2006, you are not using it anymore. Papers on MapReduce and GFS led to the open-source implementations Hadoop and HDFS. But meanwhile, Google was moving on because you were realizing that writing convoluted MapReduce code and maintaining and managing clusters was hard. So, you developed better ways to do things: Dremel, for example, is essentially SQL queries, and Colossus is a file system that we don't publicly explain that enables extremely high-throughput reads. Dremel and Colossus are offered on Google Cloud Platform as BigQuery and Google Cloud Storage, so you can use Google's second generation of our Big Data stack.

Of course, it is not just Google. The entire industry has recognized this, and that is why are you seeing a third wave of cloud.
Where the second wave of cloud was about rented infrastructure, similar to what we did in the previous lab, the third wave of cloud is about fully elastic services. The second wave of cloud is great if you want to migrate your legacy applications, but for new projects, use the third wave of cloud immediately.

Demo: Query large datasets in seconds

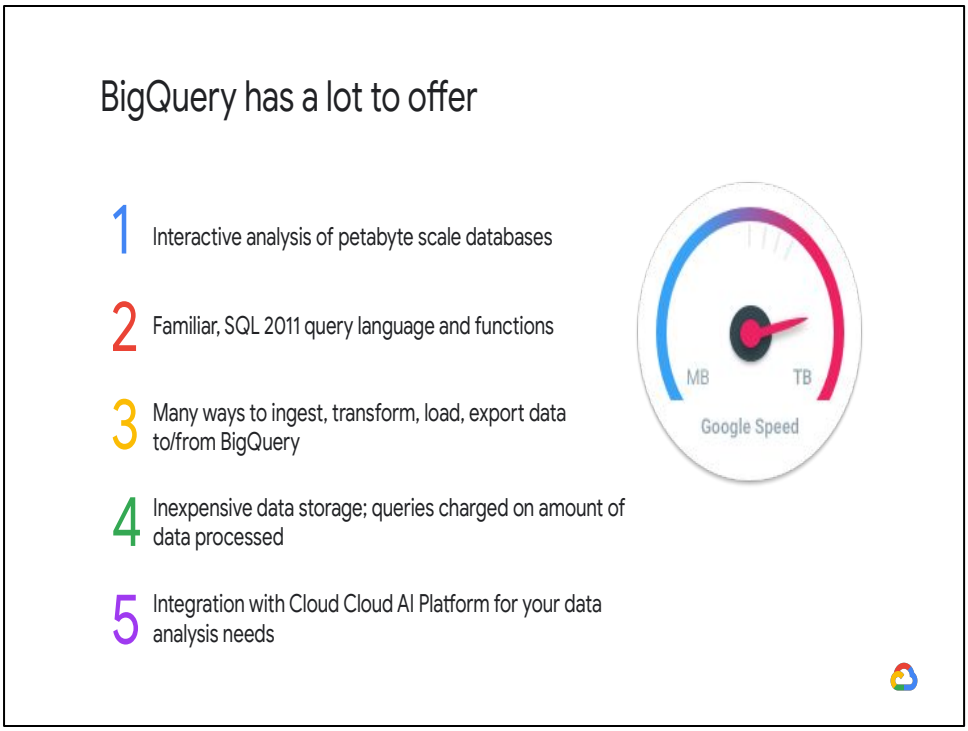
```
#standardsql

# medicare claims in 2014
SELECT
  nppes_provider_state AS state,
  ROUND(SUM(total_claim_count) / 1e6) AS
total_claim_count_millions
FROM
  `bigquery-public-data.medicare.part_d_prescriber_2014`
GROUP BY
  state
ORDER BY
  total_claim_count_millions DESC
LIMIT 5;
```

Row	state	total_claim_count_millions
1	CA	116.0
2	FL	91.0
3	NY	80.0
4	TX	76.0
5	PA	63.0

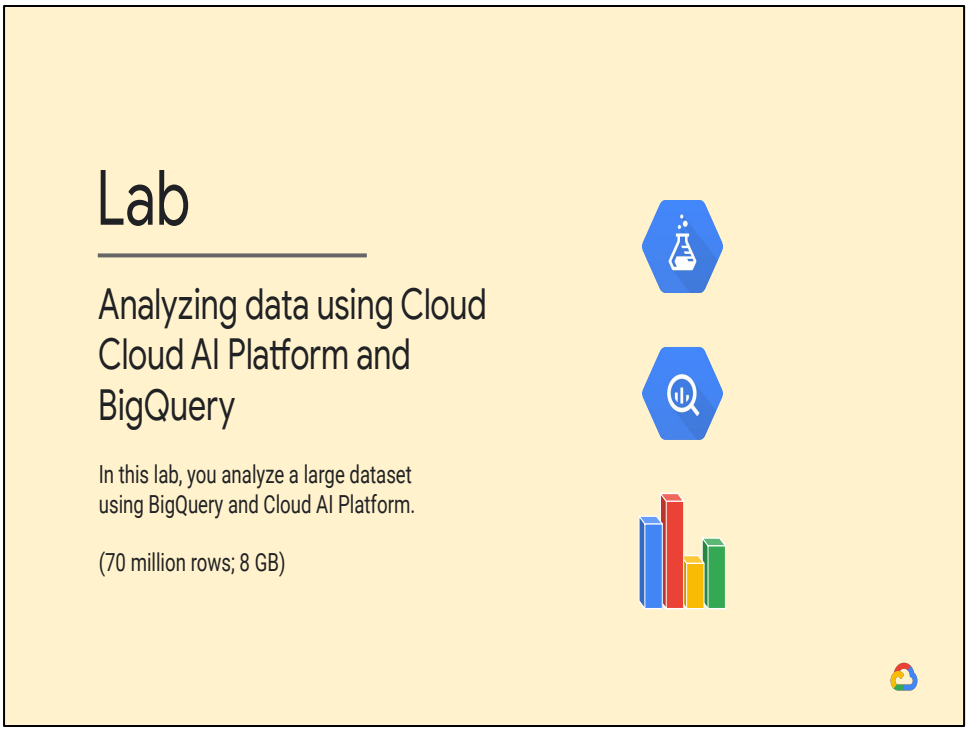
You are now on the BigQuery web console and putting in an SQL query. Essentially, you want to find healthcare insurance claims by state. You can write an SQL query and simply execute it. It's an ad hoc query. There is no need to create indexes.

A few seconds later, you get back the result. Turns out California and Florida have the highest number of claims. How big was this table? Let's go and check the Details tab and you see that it's about 24 million rows. You have successfully queried petabytes of data in BigQuery. It works because, behind the scenes, BigQuery uses thousands of machines to carry out the processing. Thousands of machines for a few seconds. This is what you referred to as the third wave of cloud.



In the quick demo, you saw that you can query large datasets interactively. In an ad-hoc manner.
The query itself was SQL 2011. A standard.
Getting data into BigQuery is quite flexible. You can use anything from uploading files from the web GUI to staging them on Cloud Storage and pointing at them, to streaming data into BigQuery. You can export from BigQuery quite easily, because of the variety of APIs. Run a SQL query and save the result in the format that you want.
Storage is quite cheap, essentially similar to Cloud Storage.
Finally, and key for our purposes, Cloud AI Platform integrates nicely with BigQuery, so you can explore, run a query, export into a Pandas dataframe, and plot it using Python.

In fact, that is exactly what you are going to do next.



In this lab, you will get to employ a pretty useful pattern: you will use BigQuery to calculate useful aggregates (percentile values and the like) over 70m rows. The result will go into a Pandas dataframe of a dozen rows.
You can then happily use that in-memory dataframe in visualization.
This is the kind of thing that would take you hours if you did it any other way. Yet, you will create the graphs in seconds.
It is important to get this kind of interactive development workflow down. Otherwise, you will not be able to work with large datasets easily.

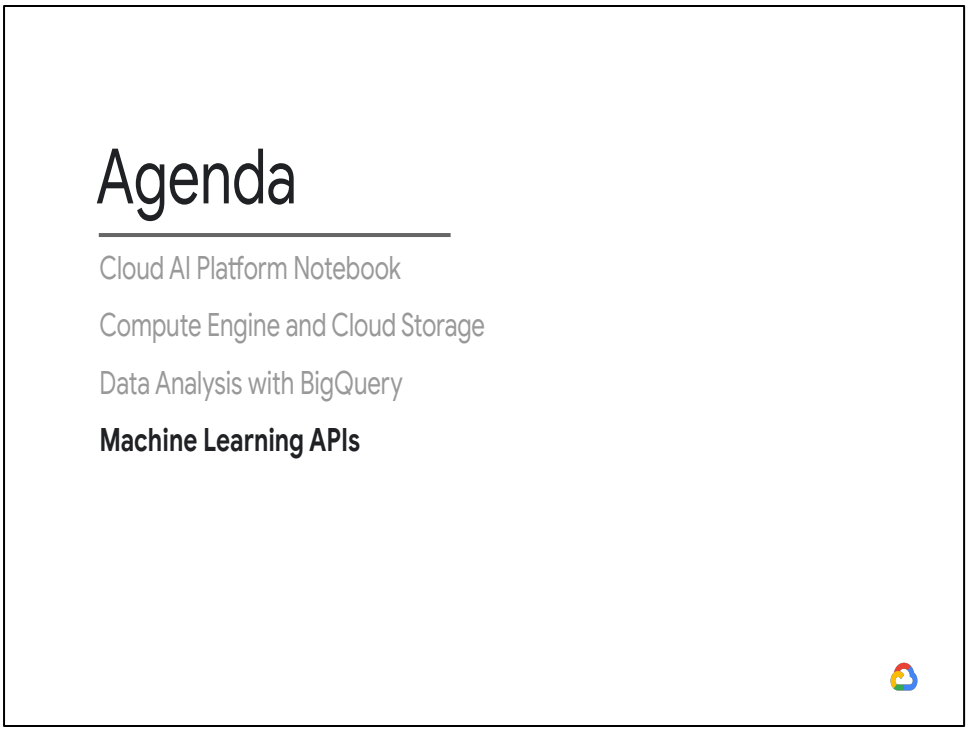
You might think that you can simply sample the dataset and work with it. However, that is a bad practice in machine learning.
One thing I like to say is that the key difference between statistics and machine learning is how we deal with outliers. In statistics, outliers tend to be removed. In ML, outliers tend to be learned. That is because the datasets in ML are typically much larger.

Link to lab:
Qwiklabs: https://googlecloud.qwiklabs.com/labs/1941/edit

No git repo. Not a Jupyter Notebook, a “step-by-step” lab.

Lab Steps

1. Launch Cloud Cloud AI Platform.
2. Invoke a BigQuery query.
3. Create graphs in Cloud AI Platform.



If you're new to machine learning, an easy way to add ML to your apps is to take advantage of pre-trained models. So, before we dive into building custom models, let's take a step back and see how we'd perform image classification *without* a ML model.

Many basic programming problems are “impossible” to solve without machine learning. For example, if you were to write code to tell the difference between an apple and an orange, what sort of properties would you look for in your algorithm (probably a series of if statements)? “Color” or “presence of a stem” are good answers here - if you looked at the color of the majority of the pixels in the image you could output red for an apple and orange if the majority of the pixels were orange. This approach would work well for these specific apple and orange images, but what if the images changed slightly?

Would those rules work if someone gave us the grayscale images?
This is the problem with rules, and this is what is meant when it is said hand-written rules don’t scale.
Instead, you want to use machine learning, supplying the model with many examples of apples and oranges, and have the system learn how to tell the difference. If you had grayscale apple/orange images, we’d have to start all over again. In this case we could look at texture.

What if you got crazy and added a third fruit to the equation (a mango)? Then you’d have to write your algorithm from scratch all over again. But these images are all relatively circular fruits and look similar. The image classification should be easier if you have images that look nothing alike, right?

Image sources:

- Apple: https://commons.wikimedia.org/wiki/File:Apple_in_lightbox.png

- [CC-BY-SA-2.0]
- Orange: https://pixabay.com/p-1218158/?no_redirect
- Mango: https://pixabay.com/p-1218147/?no_redirect

Many fruits look similar, but what about two visually distinct objects, like a dog and a mop? Those should be easy to distinguish, right?

Not so fast! These are pictures of sheepdogs and mops, and it's actually kind of hard to tell the difference.
In fact, even for machine learning systems, it can be hard. And the way that you can get good machine learning systems is that you need a really large dataset. And you need to train a custom machine learning model with lots of pictures of dogs, including sheepdogs and lots of pictures of tools, including mops. Or do you?

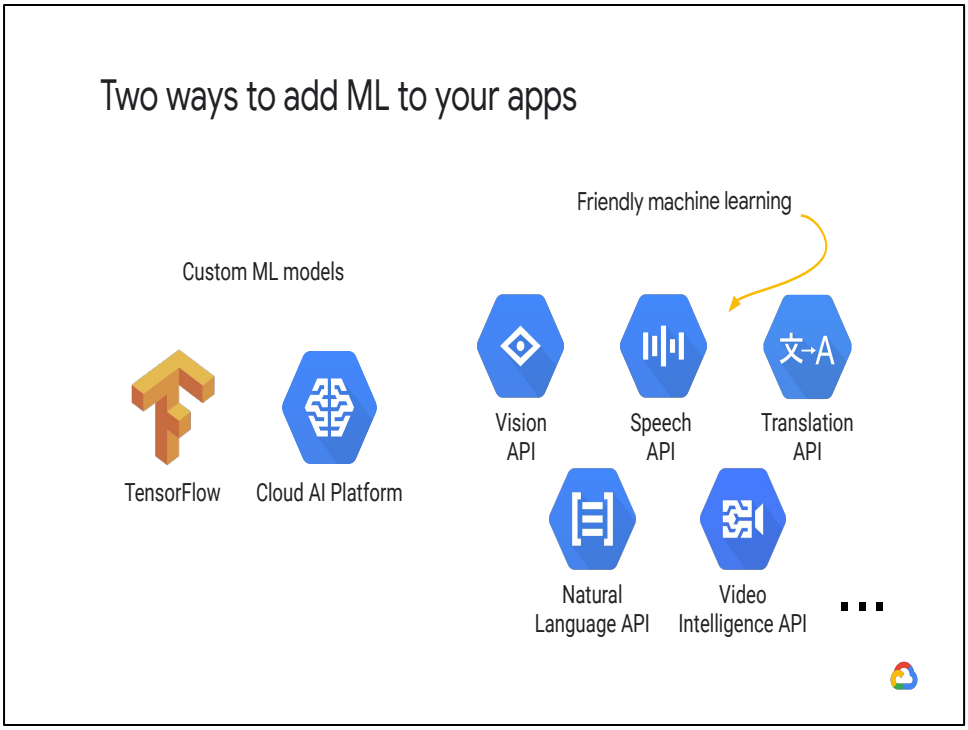
Note: we have the rights to use all these images!

- Dog: https://pixabay.com/p-1210559/?no_redirect
- Mop: https://commons.wikimedia.org/wiki/File:Mop_and_bucket.jpg

- Sheepdog photos:
  - https://commons.wikimedia.org/wiki/File:Komondor_Westminster_Dog_Show_crop.jpg [CC-BY-SA-2.5]
  - https://commons.wikimedia.org/wiki/File:2014_Westminster_Kennel_Club_Dog_Show_(12487315865).jpg [CC-BY-2.0]
  - https://www.flickr.com/photos/petsadviser-pix/16395099127 [CC-BY-2.0]
  - https://www.flickr.com/photos/denverjeffrey/6903790333

    - [CC-BY-SA-2.0]

- Mop photos:
    - https://www.shutterstock.com/image-photo/spaghetti-mop-head-cleaning-light-wooden-557398192?src=bSflYiFhnw_fuEht3mZJlQ-1-91 (purchased)
    - https://www.shutterstock.com/image-photo/red-mop-cleaning-isolated-on-white-159653360?src=gMX1f01CV3fDqa0mAEU1BA-1-2 (purchased)
    - http://www.istockphoto.com/photo/mop-and-bucket-gm166281571-23680028 (purchased)



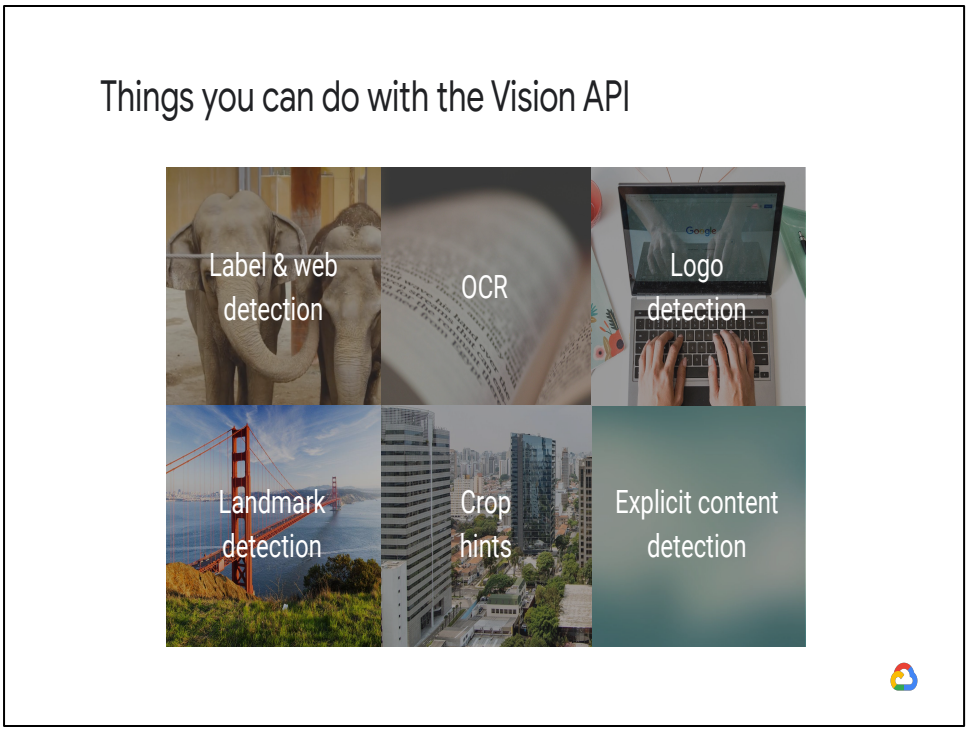
GCP has two ways for you to add ML to your apps. If you want to use custom data to build and train your own models from scratch, the Google Brain team has provided an open source library called TensorFlow. Cloud Machine Learning Engine is a way to run TensorFlow in a managed service. It allows you to train TensorFlow models in a distributed way and invoke predictions scalably.

On the right-hand side you have what you like to call “friendly machine learning.” These are APIs that give you access to a pre-trained ML model with a single REST API request. There are a variety of problems where Google exposes ML services trained with our own data. For example, if you want to transcribe speech, you could use the Speech API instead of having to collect the data, train it, and predict with it. Here you’ll get an overview of a few of these APIs along with some live demos.

GIPHY: Vision API in production

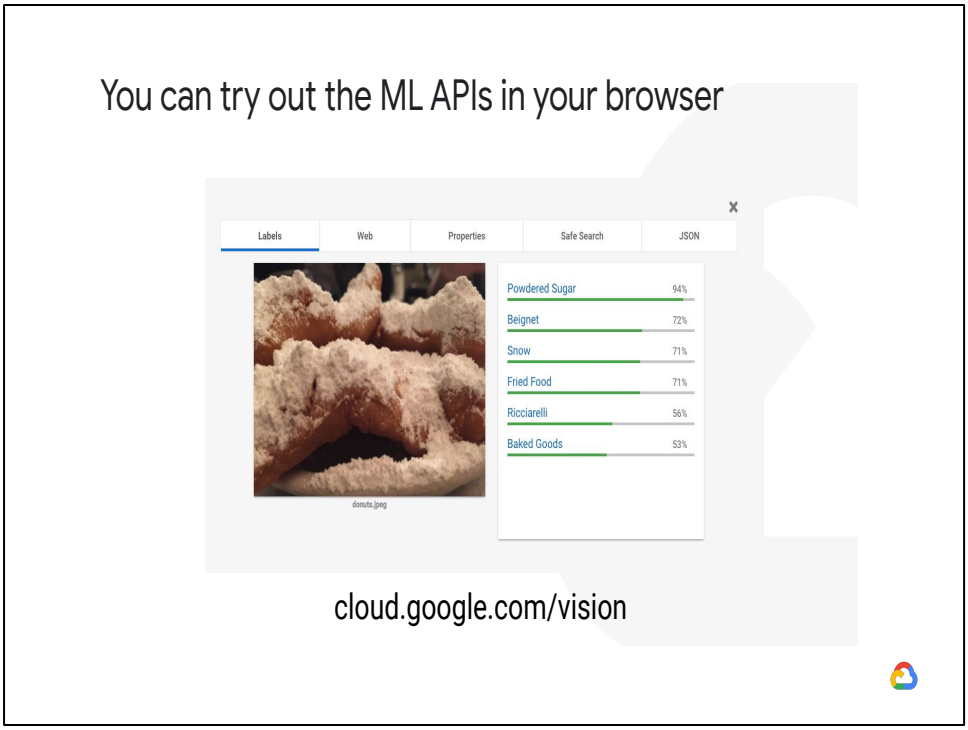
http://engineering.giphy.com

Before you talk about the features, let's see how a company is using Vision API in production. GIPHY is a company that provides an app for searching GIFs and sharing them on various social channels. They started using the Vision API's OCR feature to search for text within their images, which has significantly improved the accuracy of their search.

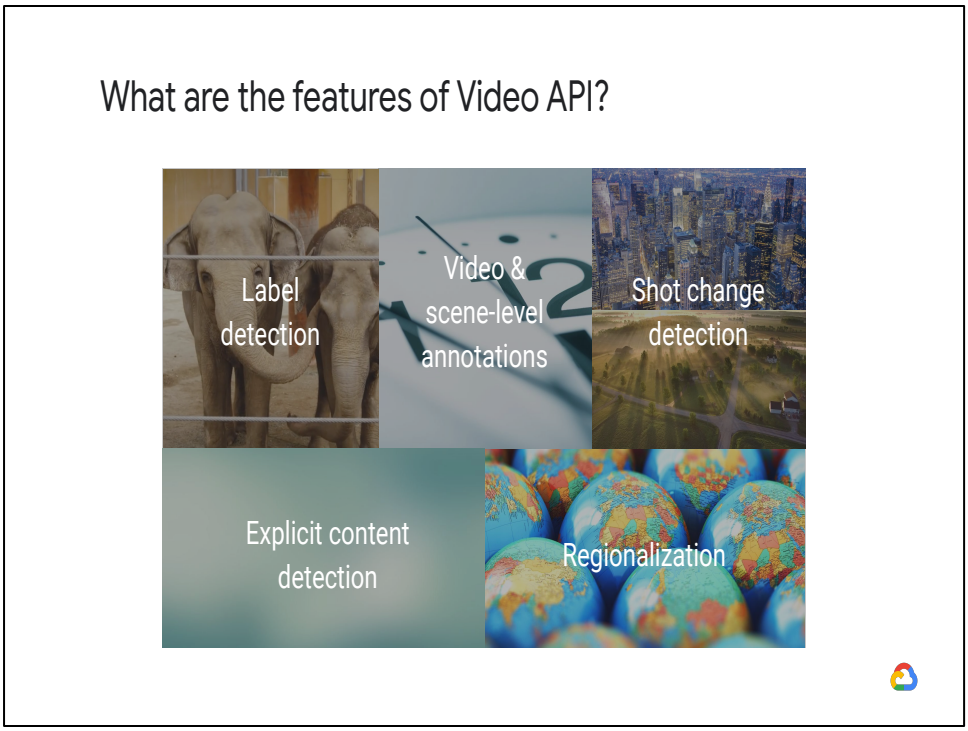


Here are some of the things you can do with the Vision API:

- Label + web: answers the question “what is this a picture of?”
- OCR: GIPHY use case. Extract text from your images
- Logo: find company logos in your images
- Landmark: identify landmarks in an image, along with latitude/longitude
- Crop hints: get suggested crop dimensions for your photos
- Explicit content: identify whether an image is appropriate or not

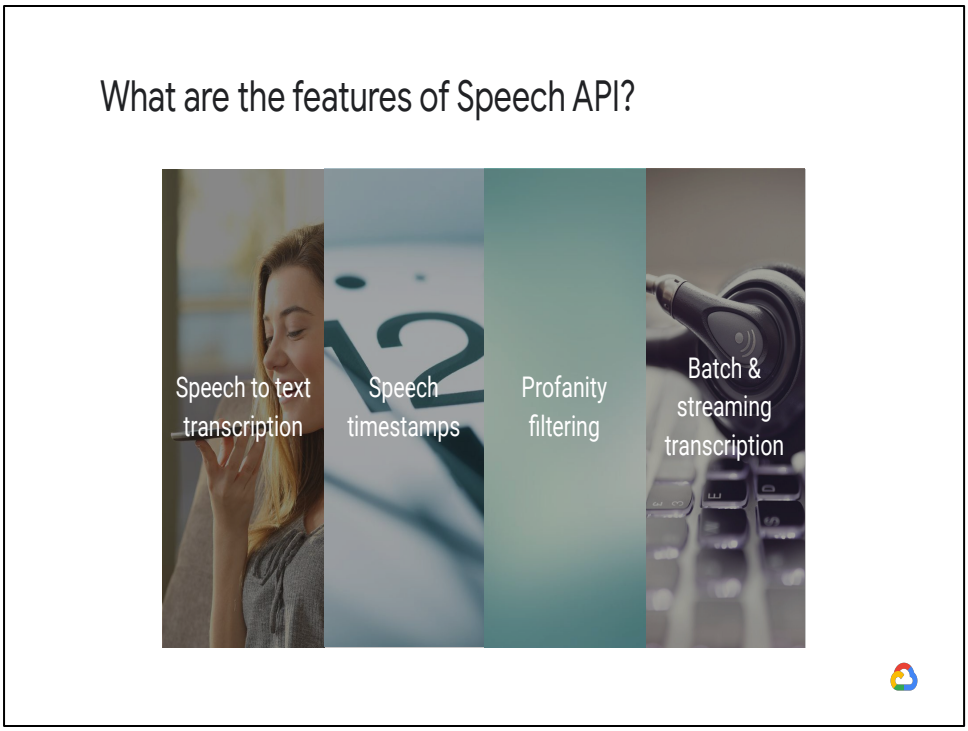


You can try out all of Google's ML APIs in the browser on their product pages. To try the Vision API, you can upload images directly in the browser before you start writing code to see what you'll get back in the Vision API response. Let's try it out!



What are the features of the Video API? At it's core it can do label detection at various levels of granularity. It will tell you what's happening in a single frame of your video, *and* it can tell you at a higher level what your video is about. In addition, it can give you the timestamps of every point in your video where the scene changed (for example, moving from a landscape to a close-up of a person).

Similar to the Vision API, it can identify inappropriate scenes in your video. You can also specify which region you'd like your Video API request to be executed in.

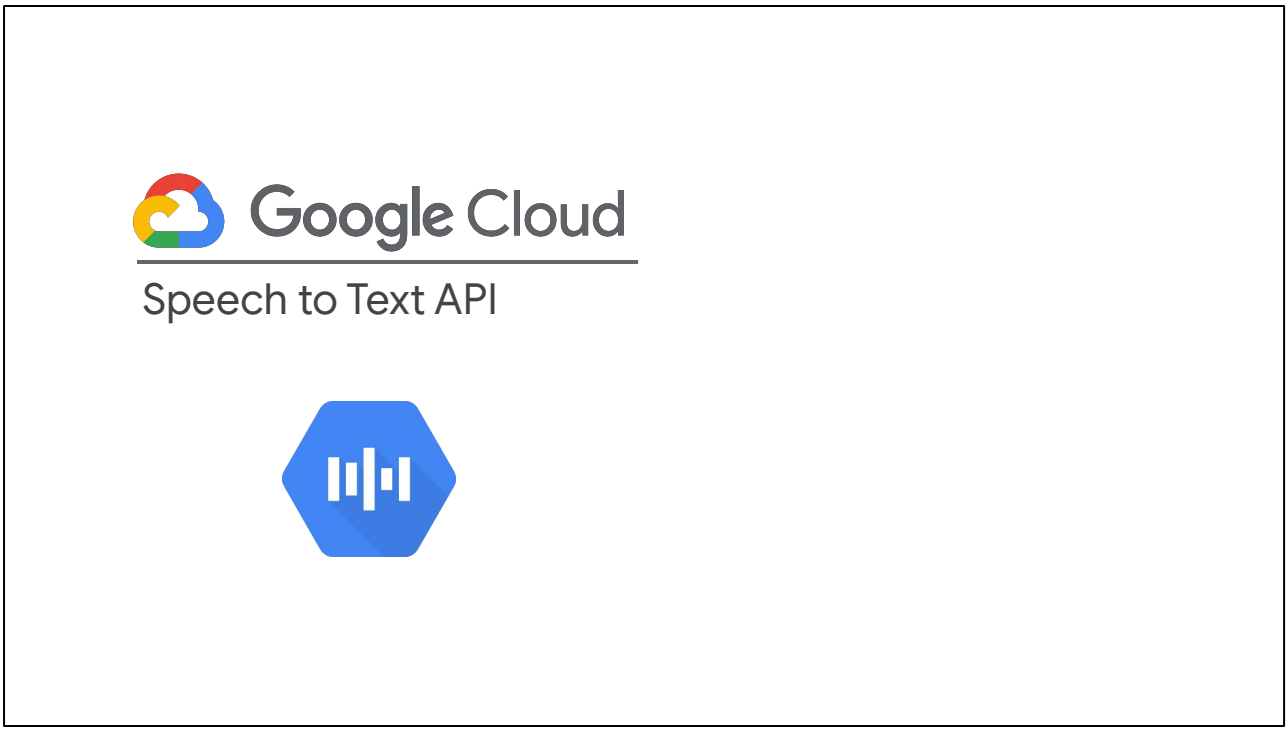


At its core, the Speech API lets you perform speech-to-text transcription. It takes an audio file as input, and outputs a text transcription of that audio. It works in 100+ languages.

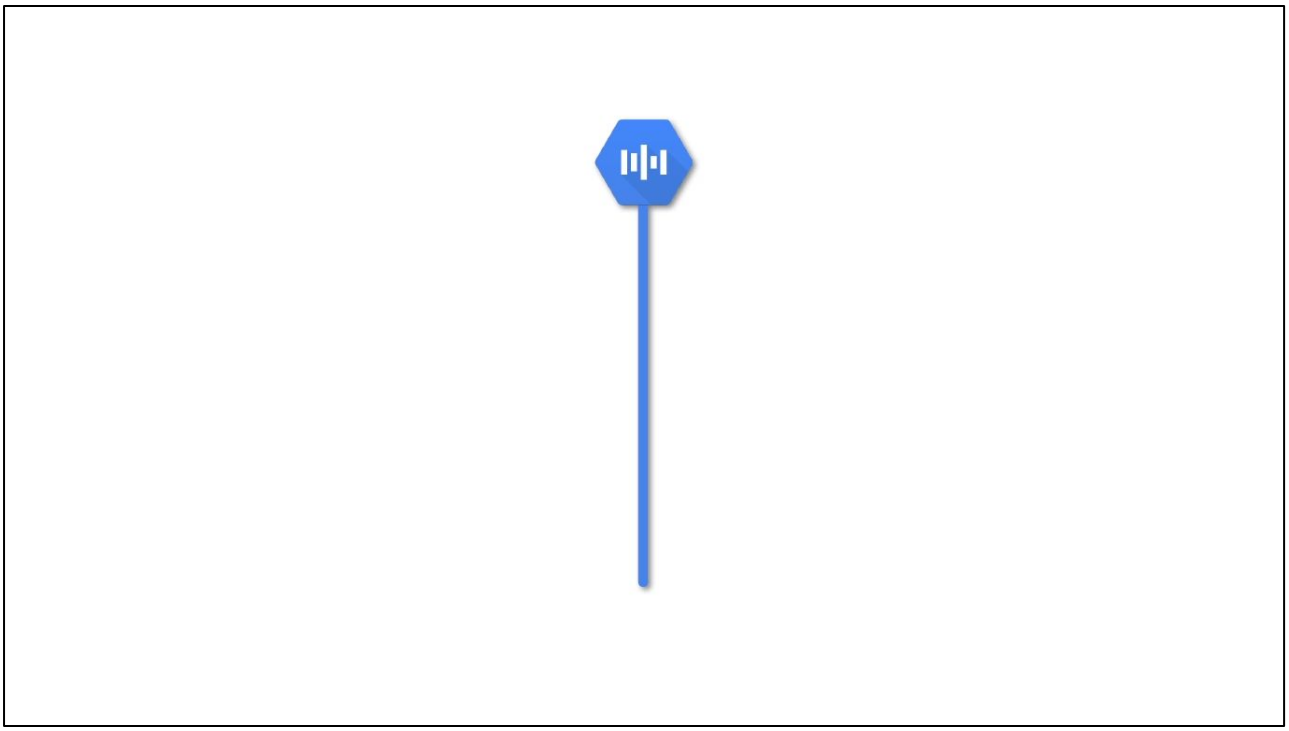
What if your application wants to let users search for a specific word or phrase within an audio snippet? You can do that with the timestamps feature. This will return the start and end time for each word in your audio file, making it easy to jump to a specific point in the audio.

Similar to Vision and Video, the Speech API will replace profanity with asterisks.

In addition to being able to process a complete audio file, the API can also help you transcribe a continuous stream of audio. It'll output transcriptions as audio comes in.

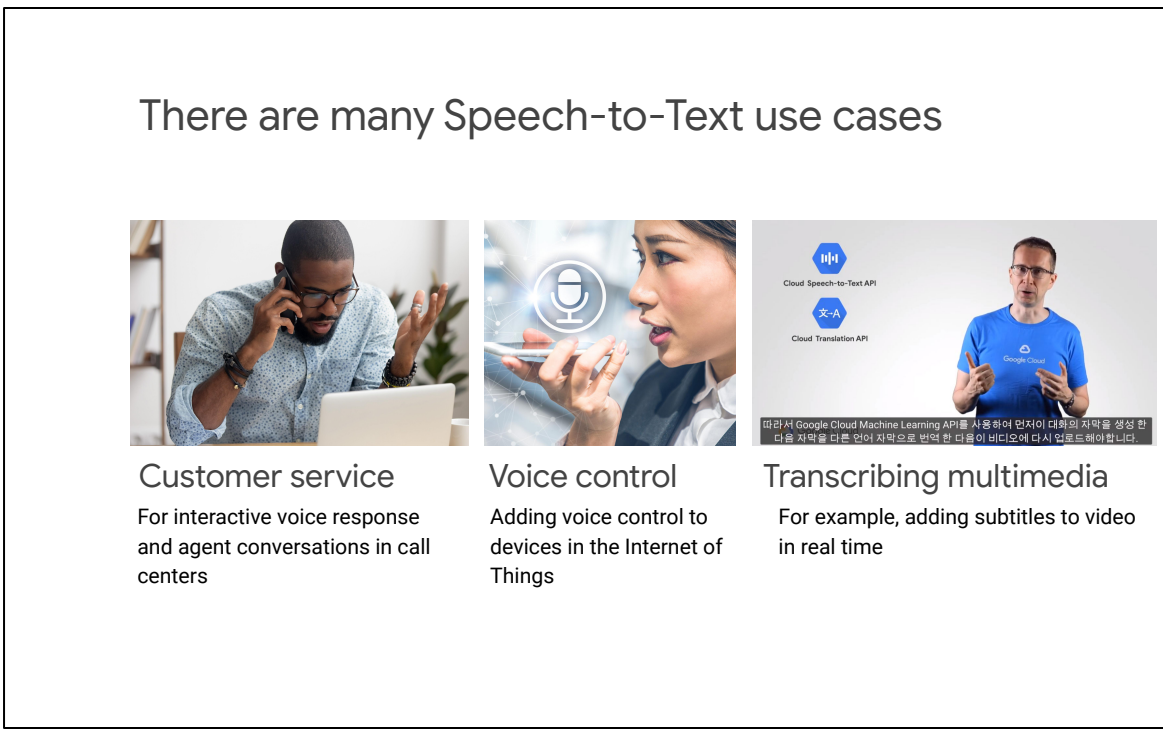


[Slide 1]
Cloud Speech-to-Text is an API that lets you convert audio to text for 71 languages in 127 variants.



[Slide 2]

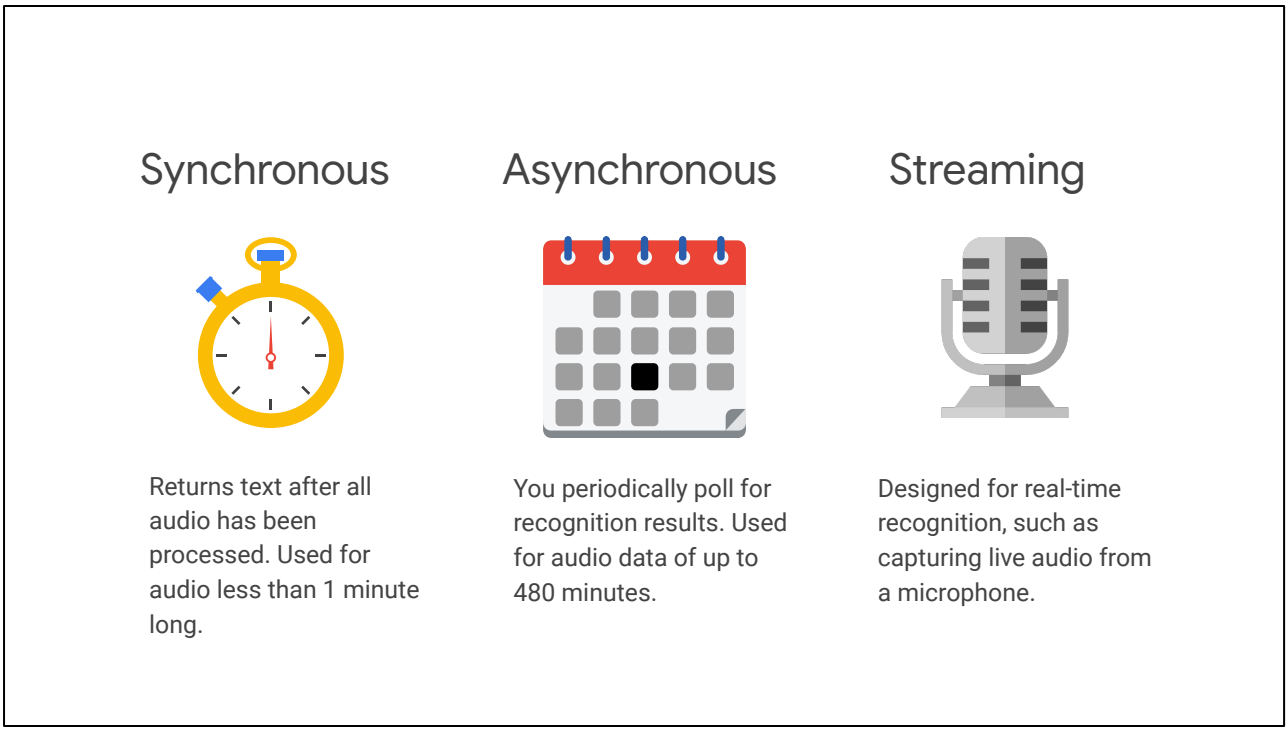
You pass it an audio file, and it returns a text transcription of that file.



[Slide 3]

Here are three of the many ways that the Speech-to-Text API gets used:

- In customer service, for interactive voice response and agent conversations in call centers;
- Adding voice control to devices in Internet of Things applications, and
- Transcribing multimedia, for example adding subtitles to video in real time.

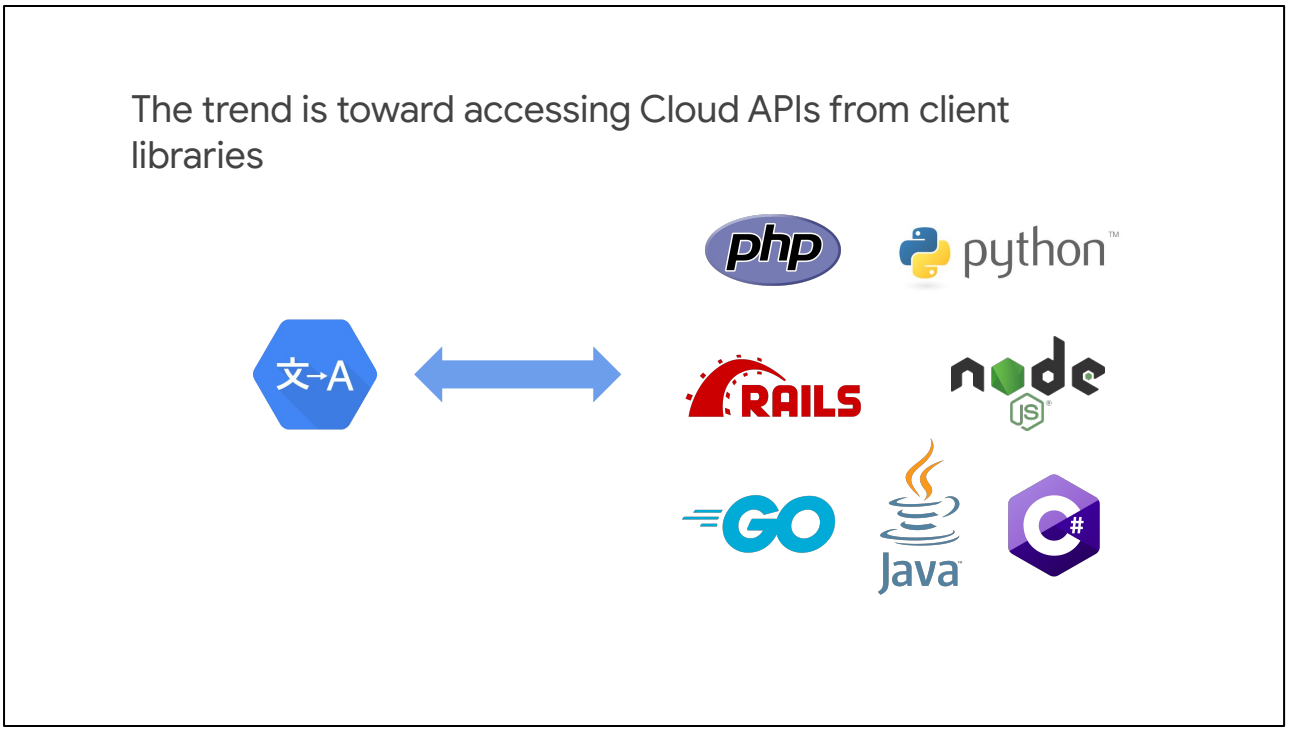


[Slide 4]

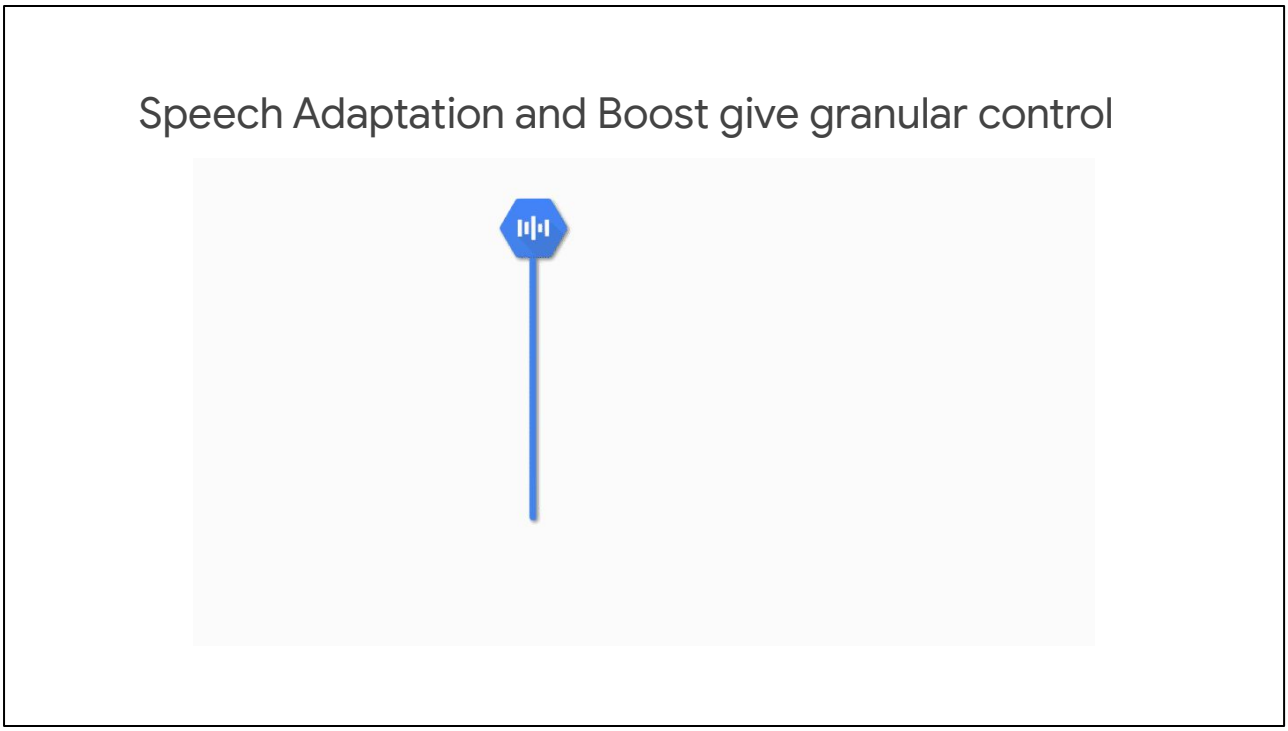
Speech-to-Text has three main modes for processing short, long, and streaming audio:

- **Synchronous:** You send audio to the Speech-to-Text API, and it returns text after all audio has been processed. This method is used for audio less than 1 minute long.
- **Asynchronous:** You send audio to the Speech-to-Text API, and it initiates a *Long Running Operation*. You periodically poll for recognition results. The asynchronous method is for audio data of up to 480 minutes. And,
- **Streaming:** This is designed for real-time recognition, such as capturing live audio from a microphone. Streaming recognition allows results to appear, for example, while a user is still speaking.

Synchronous and Asynchronous translation use the REST or gRPC APIs; Streaming uses gRPC only.



[Slide 5]
We are showing you how to access the APIs through Client URL requests (CURL). There is an increasing trend to access Google Cloud APIs from programming language client libraries. All the documentation to do this is available on the Google Cloud site.



[Slide 6]

Although Google's natural language machine language processing is excellent at understanding context, you can give it hints using Speech Adaptation.

Domain-specific terms, rare words, and words that sound the same can all be clarified by passing words and phrases in the SpeechContext object in the JSON package that accompanies your audio. For example, to increase the probability that Speech-to-Text recognizes the word "bye" rather than "buy" when it transcribes your audio data, pass "bye" in the `phrases` field of a SpeechContext object.

When you provide a multi-word phrase, Speech-to-Text is more likely to recognize those words in sequence. Providing a phrase also increases the probability of recognizing portions of the phrase, including individual words.

In practice, speech adaptation provides a relatively small effect, especially for one-word phrases. However, you can boost this effect by assigning more weight to some phrases than others. In our example, your audio may reference "bye" and "buy," but the meaning 'Goodbye' is more relevant than 'to purchase.' No problem: apply a boost value to BYE so that the machine model will favor the word BYE over BUY.

Finally, Classes are used by Speech-to-Text to identify and treat groups of

words in the same way; for example, by automatically converting spoken numbers into addresses, years, and currencies.

Together, these features give you granular control over the speech model.

Additional features improve output quality

1	**Auto-detect**	Specify up to four languages
2	**Diarize speakers**	Distinguish between speakers
3	**Automatic punctuation**	Use punctuation to mimic how something is said
4	**Profanity filter**	Filter out inappropriate content

[Slide 7]

There are other ways you can fine tune the output:

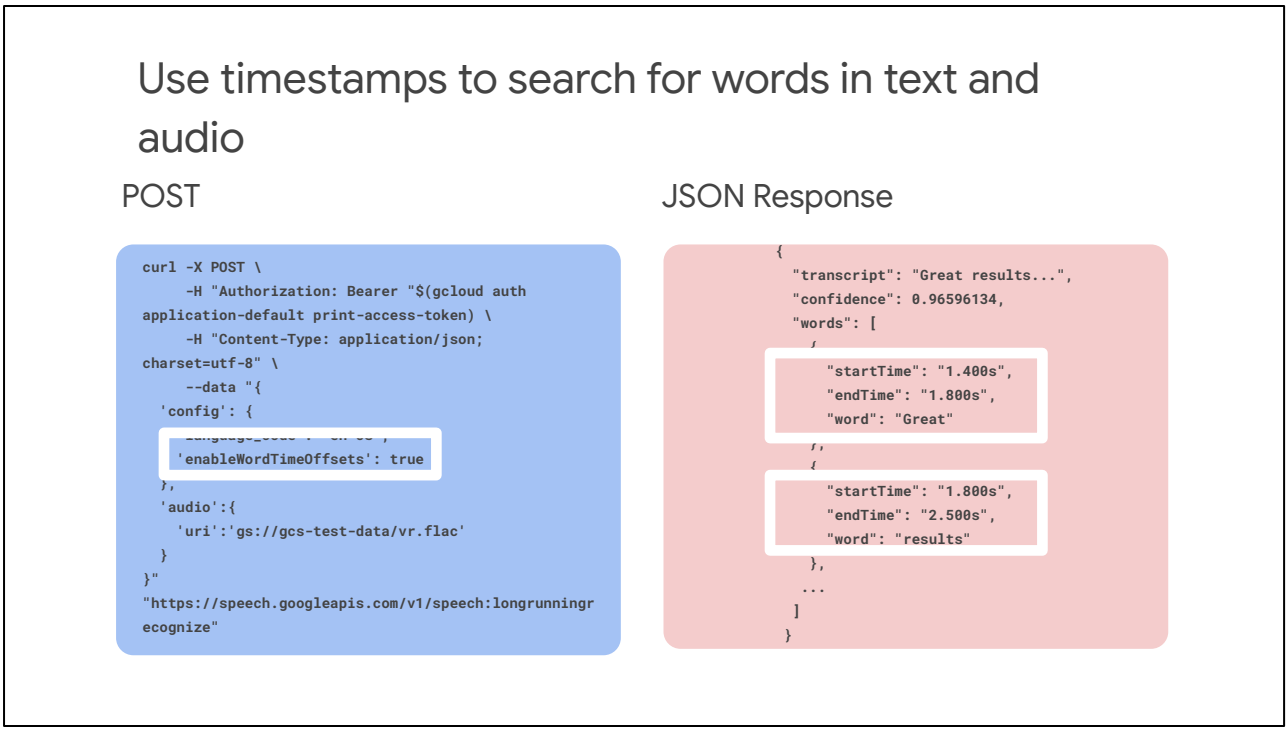
With **Auto-detect language**, you can specify up to four language codes, and Speech-to-Text will identify the correct language in multilingual audio.

Speaker diarization determines not just what was said, but who said it.

Automatic punctuation attempts to mimic how a given user might have written down what they said, with punctuation! And,

Content filtering will detect and filter out inappropriate or unprofessional content.

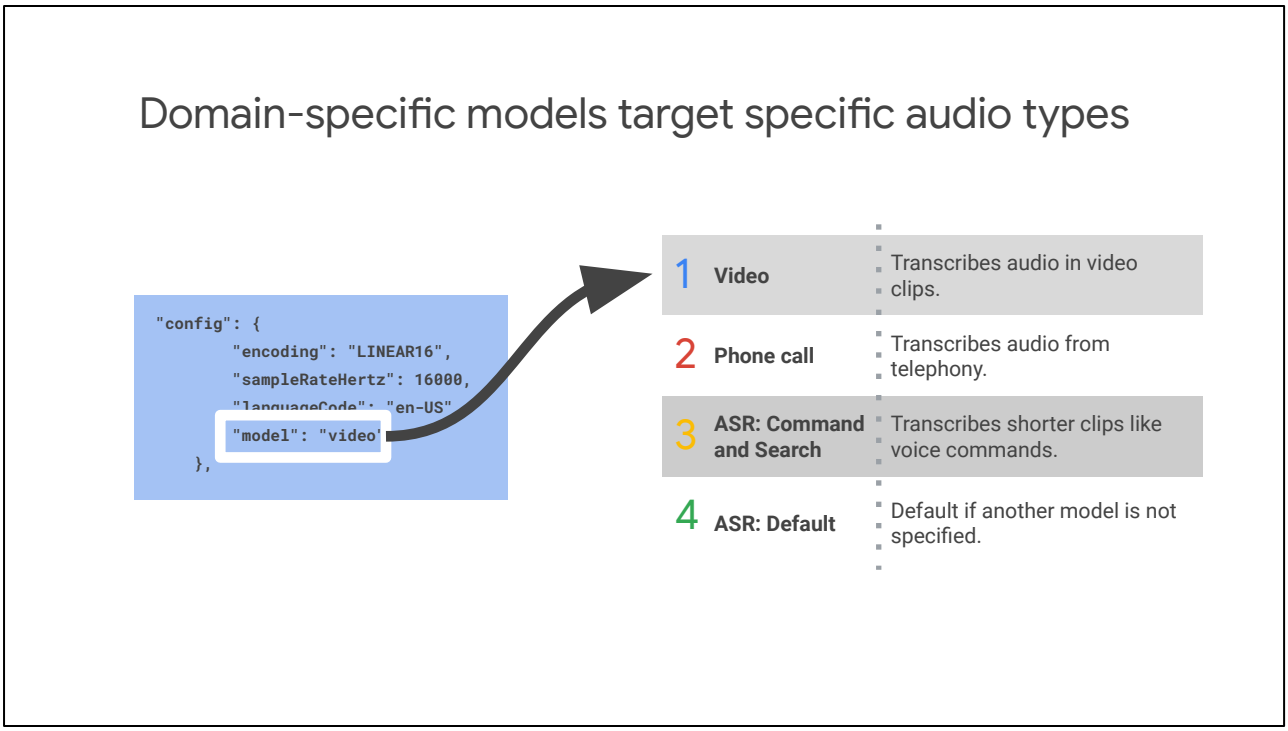
These are just some of the metadata you can supply along with your audio to give better transcription results.



[Slide 8]
Often when transcribing audio, you need to match words in the text with where they are spoken in the audio. This can be difficult with longer audio files.

The Speech-to-Text API simplifies this with time offset (timestamps), measuring for each word in the text where it is spoken in the audio.

To use this feature, set the `enableWordTimeOffsets` parameter to `true` in your request configuration. In your JSON response, you get back each word timestamped with its startTime and endTime.



[Slide 9]

You can improve transcription results by specifying the source of the original audio in the JSON package you send with your audio. There are different speech recognition models for specific audio types and sources.

The **Video model** is used for transcribing audio in video clips or audio that includes multiple speakers.

The **Phone call** model is used for transcribing audio from a phone call. This model performs up to 60% better on telephony audio than the base model.

ASR: Command and search is used for transcribing shorter audio clips, like voice commands.

And finally, **ASR: Default** is used if your audio does not fit one of the other models.

[Slide 10]

Let’s see how one company uses Speech-to-Text API to help hearing-impaired people make and receive audio phone calls.

Nagish (Hebrew for 'Accessible') lets you take phone calls using text instead of voice. Nagish is ‘app-less’; that means you can use any chat service, such as Facebook Messenger, to receive calls, and Nagish does the Speech-to-Text integration automatically.

Demo: Speech timestamps

1

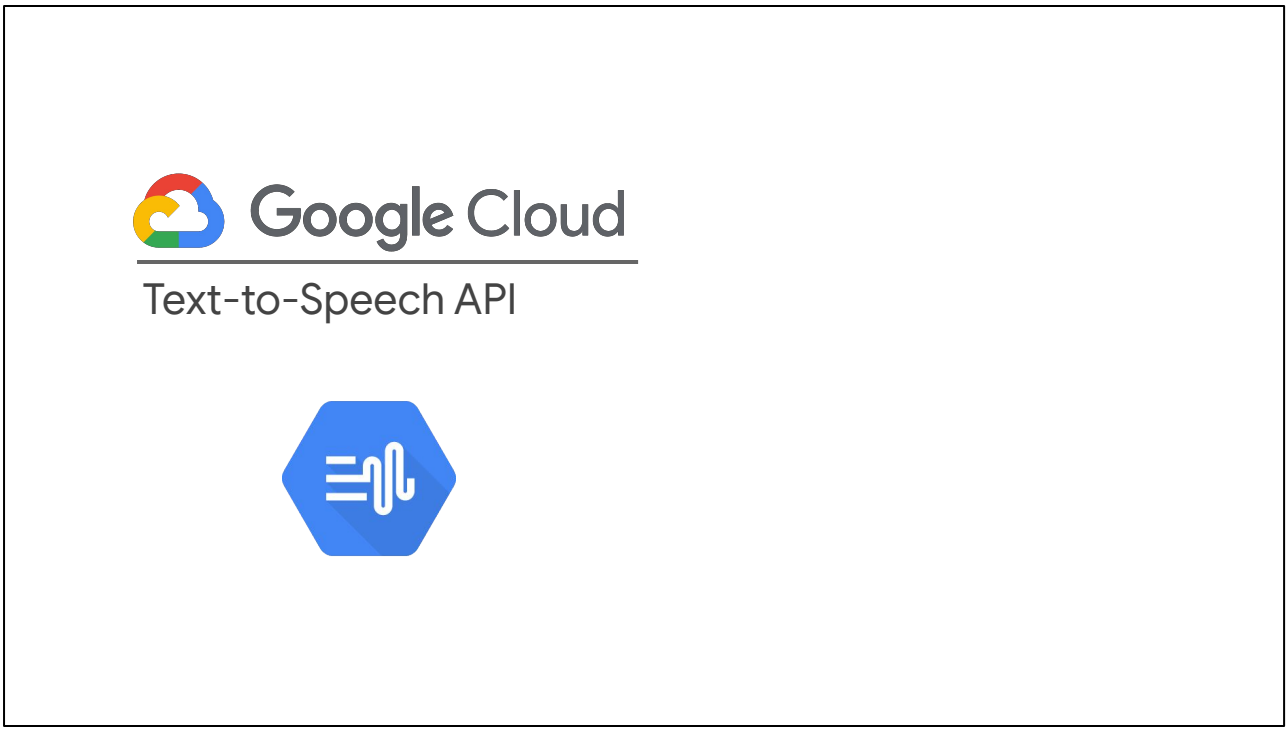
Extract audio from a video

2

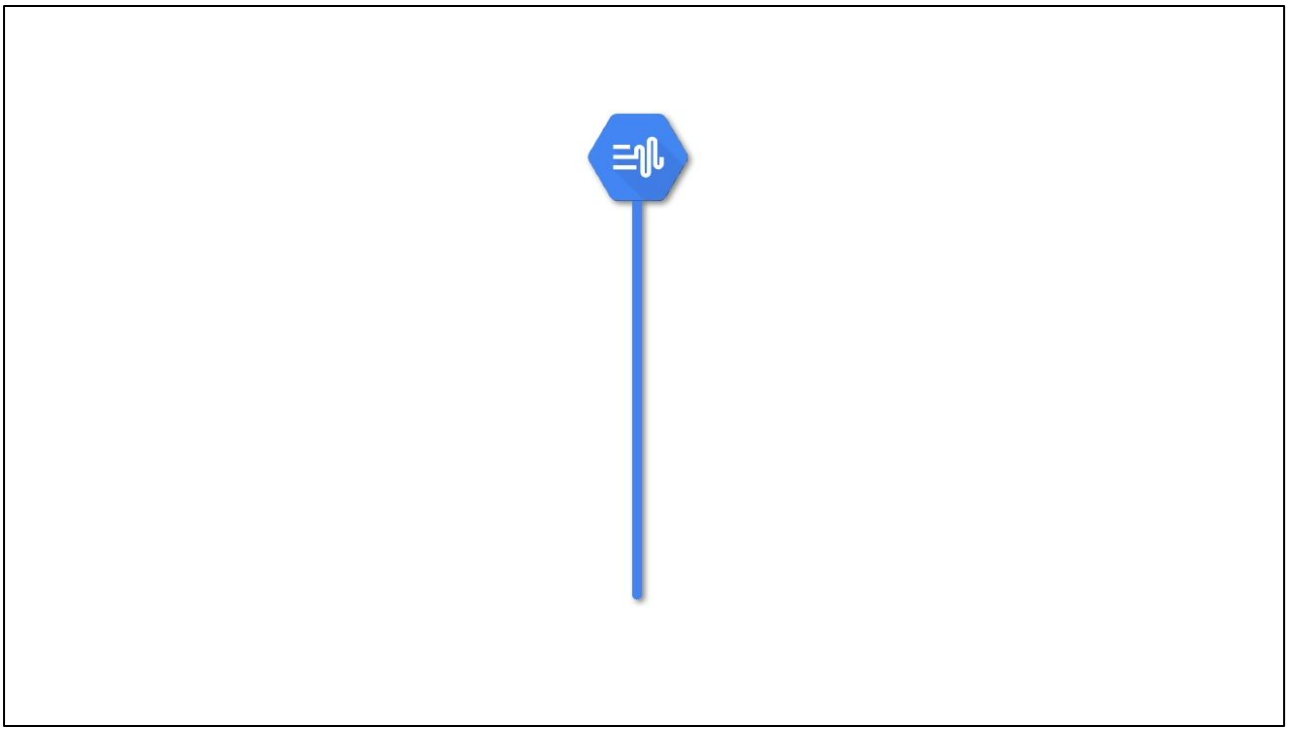
Send audio to Cloud Speech for transcription & timestamps

3

Visualize & search videos in a UI



[Slide 1]
Cloud Text-to-Speech is an API that lets you convert text into human-like speech.



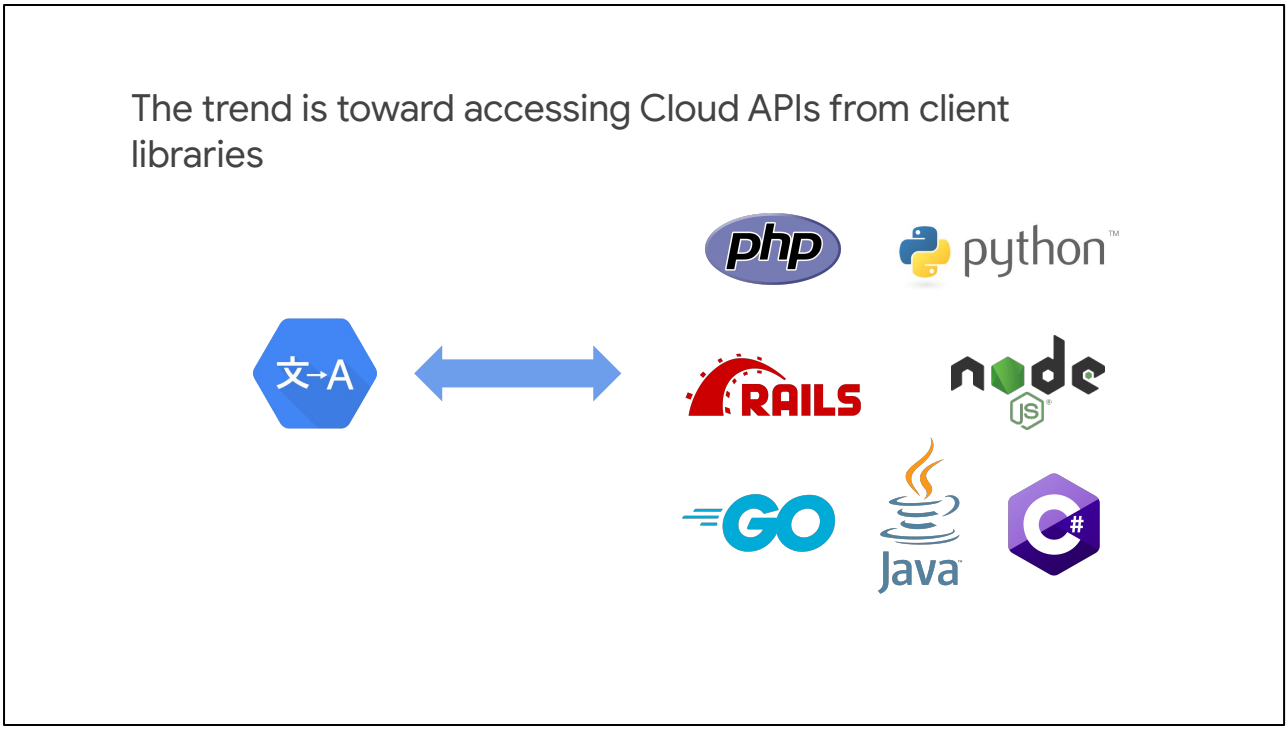
[Slide 2]

You pass it a text file, and it returns raw audio data as a base64-encoded string. You must decode the base64-encoded string into an audio file before an application can play it. Most platforms and operating systems have tools for decoding base64 text into playable media files.

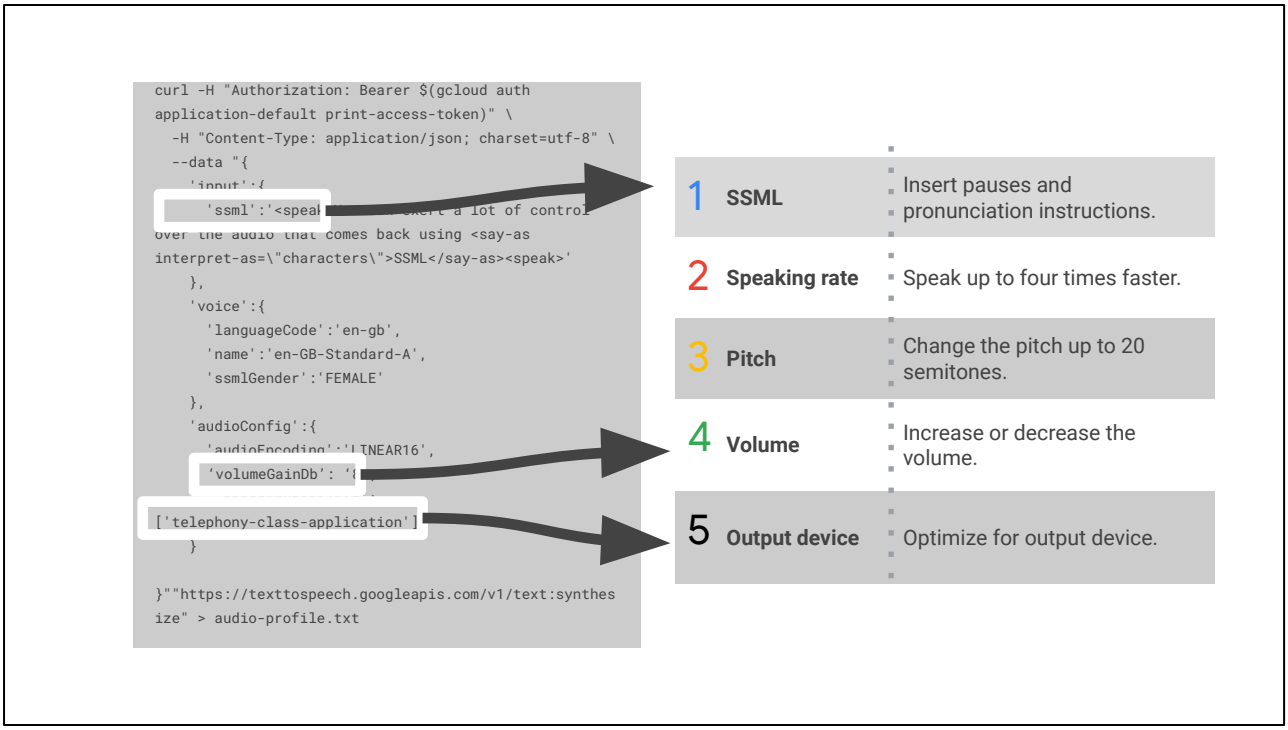
Machines that talk to us are now commonplace

[Slide 3]
You should be familiar with machines that talk to us, thanks to the Google Assistant, Google Translate, and Google Search. Text-to-Speech supports any application or device that can send a REST or gRPC request, including phones, PCs, tablets, and IoT devices like TVs and speakers.

Application developers have created life-like interactions with users that transform customer service, sales and consumer device interaction.



[Slide 4]
We are showing you how to access the APIs through Client URL requests (CURL). There is an increasing trend to access Google Cloud APIs from programming language client libraries. All the documentation to do this is available on the Google Cloud site.

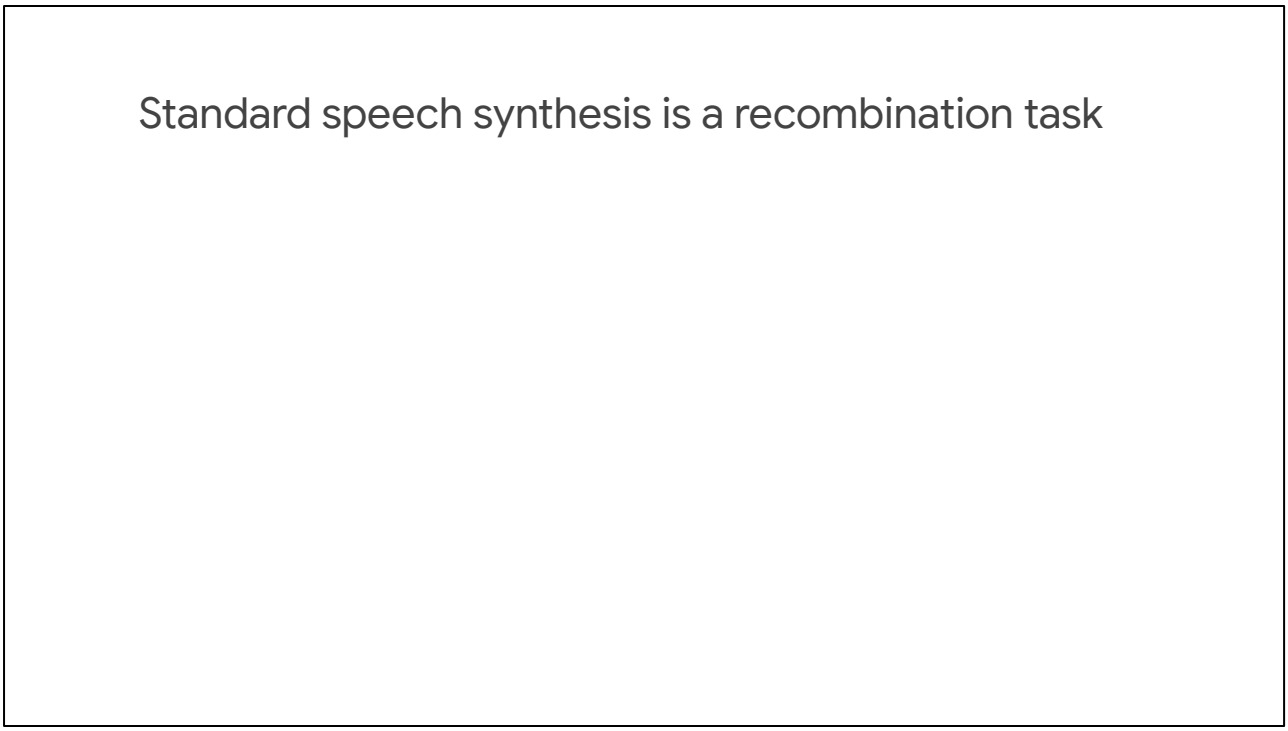


[Slide 5]

You can exert a lot of control over the audio that comes back. For example you can add pauses, numbers, date and time formatting, and other pronunciation instructions using Speech Synthesis Mark-up Language. Or you can change the speaking rate or pitch of the default voice. You can also increase or decrease the volume and optimize the audio for the output device, such as speakers or telephone playback.

[Slide 6]

The Text-to-Speech API creates raw audio data of natural, human speech. You can access more than 180 voices across more than 30 languages and variants, with new voices and languages being added frequently. The voices differ by language, gender, and accent (for some languages). When you send a request to Text-to-Speech, you must specify a *voice* that 'speaks' the words.

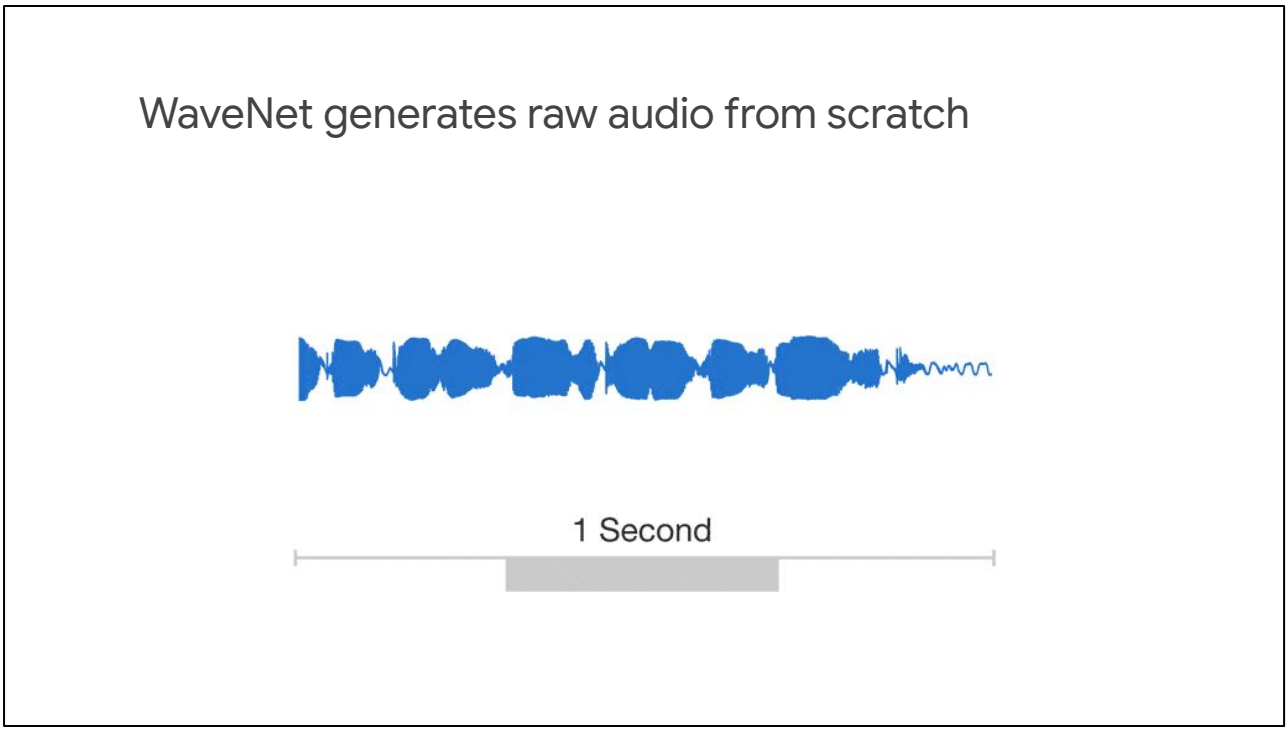


[Slide 7]

The process of translating text input into audio data is called *synthesis,* and the output of synthesis is called *synthetic speech*.

Synthesis is largely based on a very large database of short speech fragments recorded from a single speaker, which are broken into tiny chunks and then recombined to form complete words and sentences. This is called *concatenative text-to-speech*. Those tinny-sounding unnatural computer voices are generated this way. With concatenative TTS, it is difficult to modify the voice (for example, switching to a different speaker or altering the emphasis or emotion of their speech), without recording a whole new database.

Another method is Parametric TTS. This uses a series of rules and parameters about grammar and mouth movements to guide a computer-generated voice. It’s cheaper and faster, but even less natural sounding.

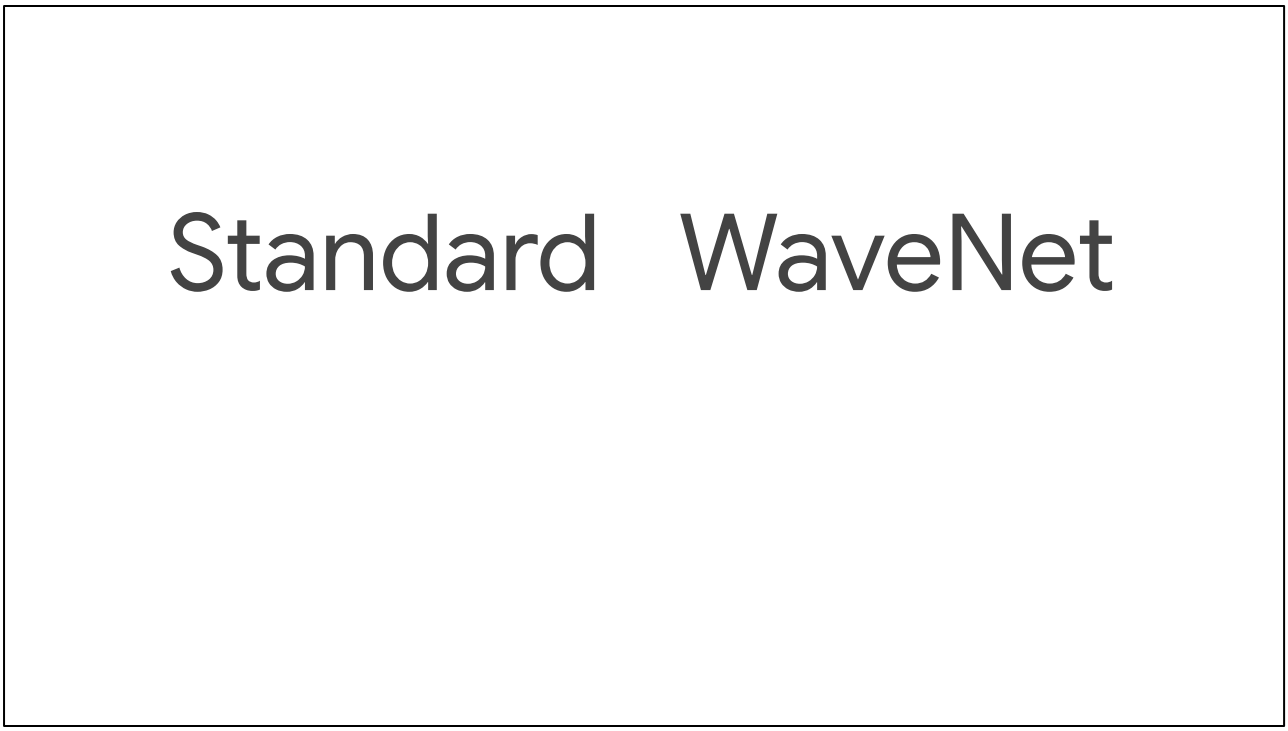


[Slide 8]

Google has taken a totally different approach with WaveNet. A WaveNet model generates more natural speech sounds by creating raw audio waveforms from scratch, with more human-like emphasis and inflection on syllables, phonemes, and words. Developers usually avoid modelling raw audio because its data points change so quickly: typically 16,000 samples per second or more, with important voice structure at very granular scale.

WaveNet was built using a neural network model similar to those used for analyzing images. During the training phase, the WaveNet model determines the underlying structure of speech, such as which tones follow each other, and which waveforms are or are not realistic. The trained network synthesizes a voice one sample at a time, with each sample taking into account the properties of the previous one. This creates natural-sounding voices with emphasis, intonation, accent, and even lip smack sounds!

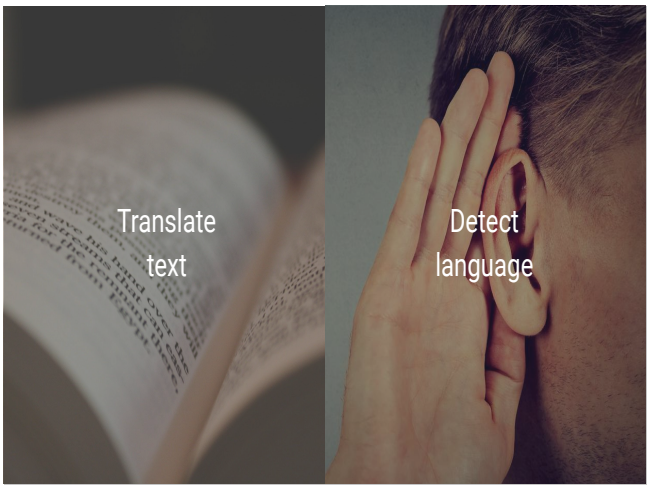
WaveNet began as a research model that was too computationally intensive for consumer products. Then it migrated to services like Search and the Google Assistant. Now the Text-to-Speech API also offers a group of premium voices generated using *WaveNet* technology.



[Slide 9]
Listen to the difference in these examples.

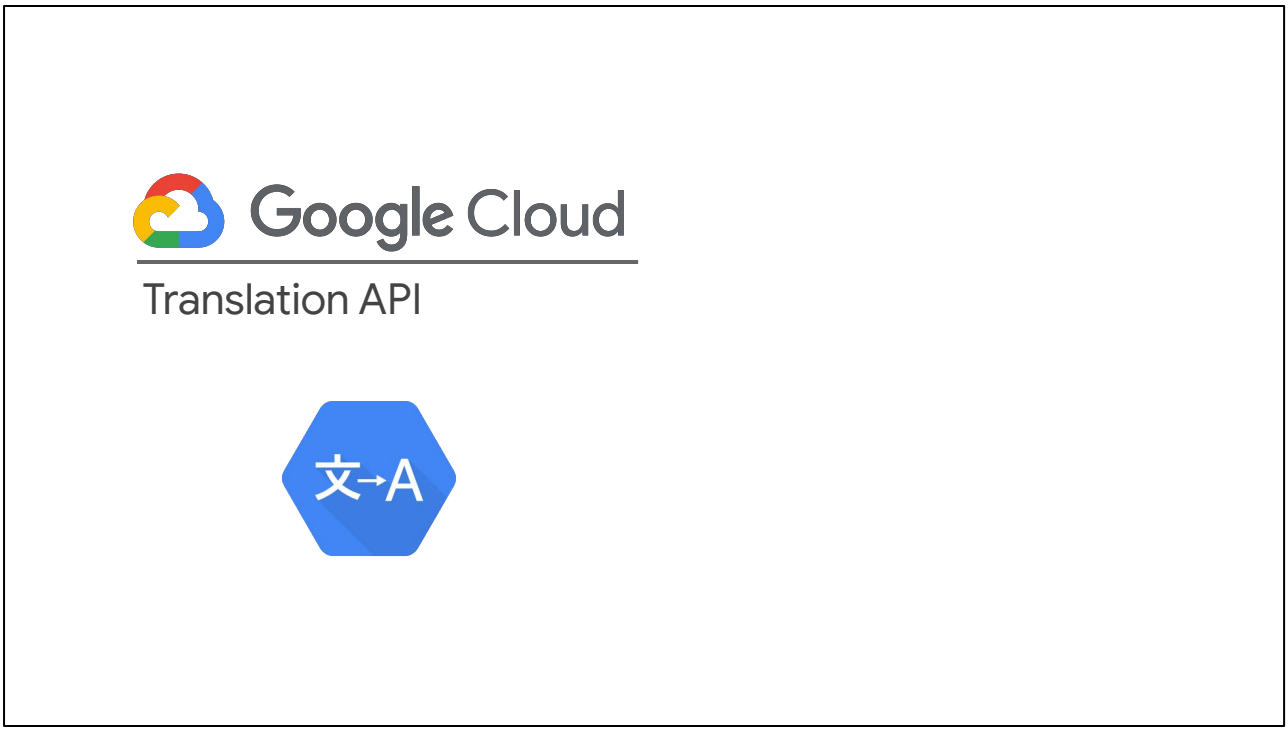
The more natural sound of the WaveNet voice is clearly distinguishable.

What are the features of Translation API?



The Translation API can translate text or simply detect the language of inputted text.

Learn more on the product page: cloud.google.com/translation



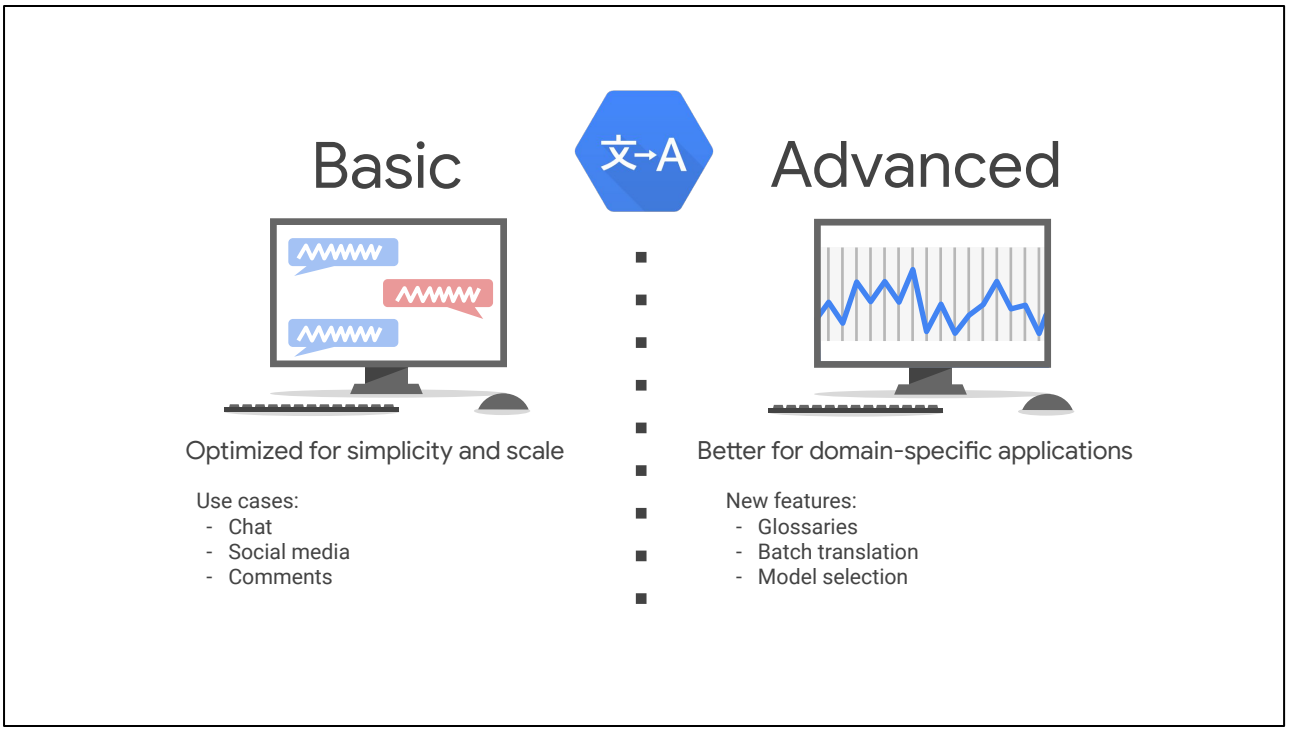
[Slide1]

Cloud Translation is an API that lets you dynamically translate between over 100 languages using pre-trained machine learning models.

[Slide2]

The purpose of the Cloud Translation API is to help make content accessible to users in their own language.

You provide the data, Google does the ML.



[Slide3]

There are two editions: Basic and Advanced. Basic is optimized for simplicity and scale, and is better for casual user-generated content such as chat, social media, and comments.

Advanced has all the capabilities of Basic plus many more features and is better for domain-specific applications; for example, where you might need legal and financial services phrases, and long-form content that is typical in localization services.

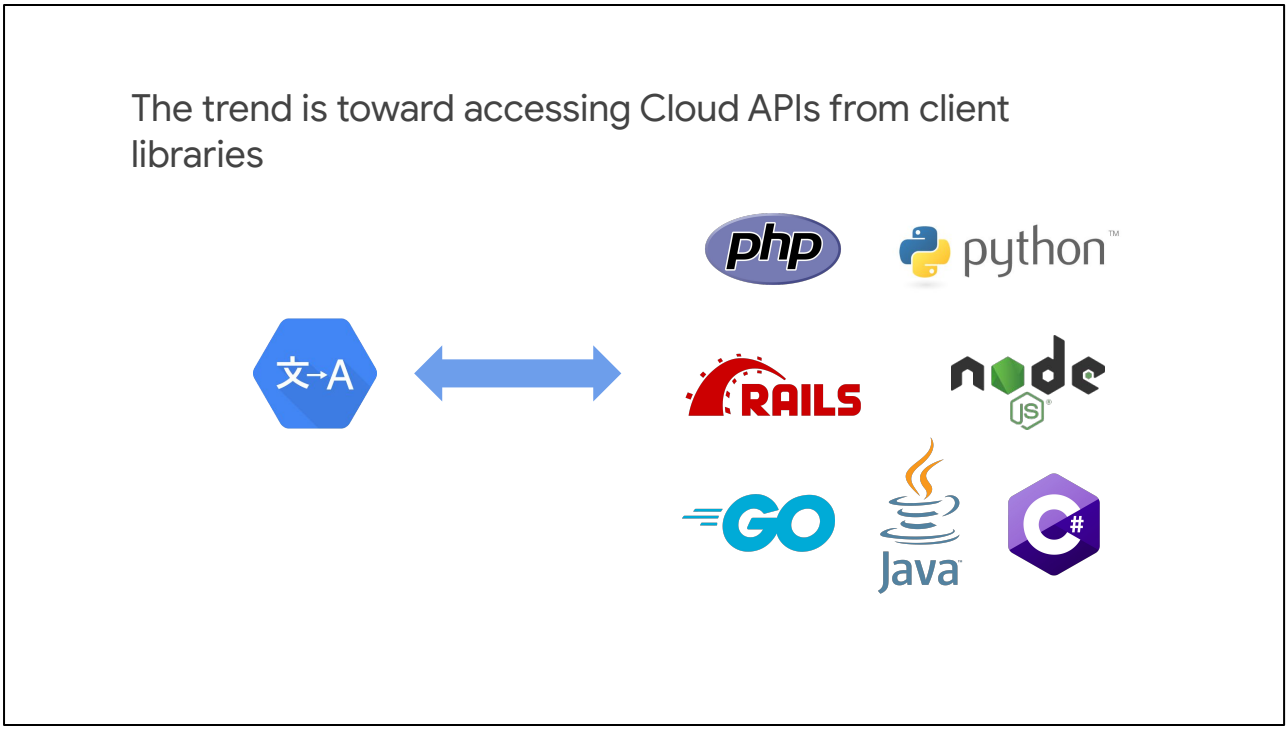
The three key new features in Advanced are:

- Support for custom dictionaries or glossaries
- The capability to do asynchronous batch translation requests, and
- The flexibility to choose which machine learning model to use, which we will look at in detail later.

If you are planning a new project, you should build your application with Cloud Translation - Advanced, to take advantage of the new features and service improvements. Cloud Translation - Basic remains available, but does not support the new features in Advanced.

Advanced uses service accounts instead of API keys, because the API

interfaces with your resources (for example, glossaries).



[Slide 4]
We are showing you how to access the APIs through Client URL requests (CURL). There is an increasing trend to access Google Cloud APIs from programming language client libraries. All the documentation to do this is available on the Google Cloud site.

Post a request, get a response

POST

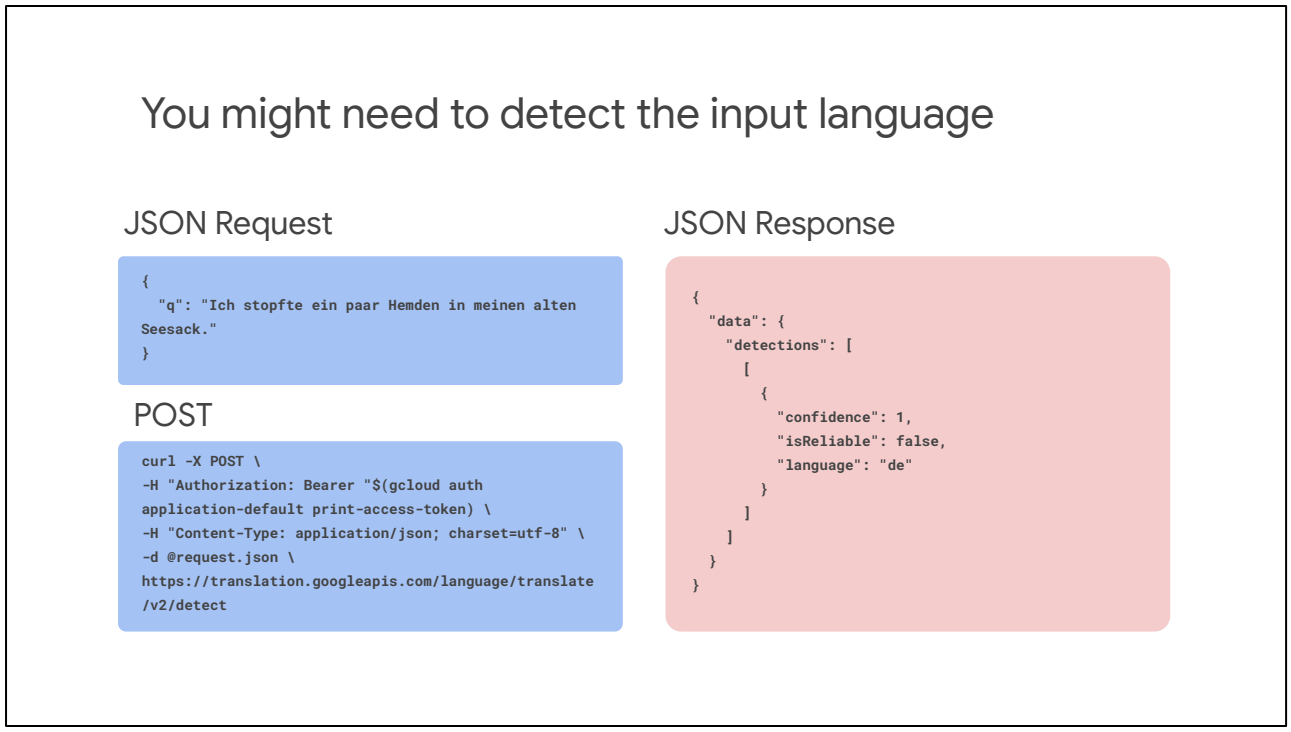
```
curl -s -X POST -H "Content-Type: application/json" \
    -H "Authorization: Bearer "$(gcloud auth
application-default print-access-token) \
    --data "{
  'q': 'I stuffed a shirt or two into my old
carpet-bag, tucked it under my arm, and started for
Cape Horn and the Pacific.',
  'source': 'en',
  'target': 'de',
  'format': 'text'
}"
"https://translation.googleapis.com/language/translat
e/v2"
```

JSON Response

```
{
  "data": {
    "translations": [
      {
        "translatedText": "Ich stopfte ein paar Hemden in
meinen alten Seesack, nahm ihn unter den Arm und machte mich
auf nach dem Kap Hoorn und dem Pacific",
        "detectedSourceLanguage": "en"
      }
    ]
  }
}
```

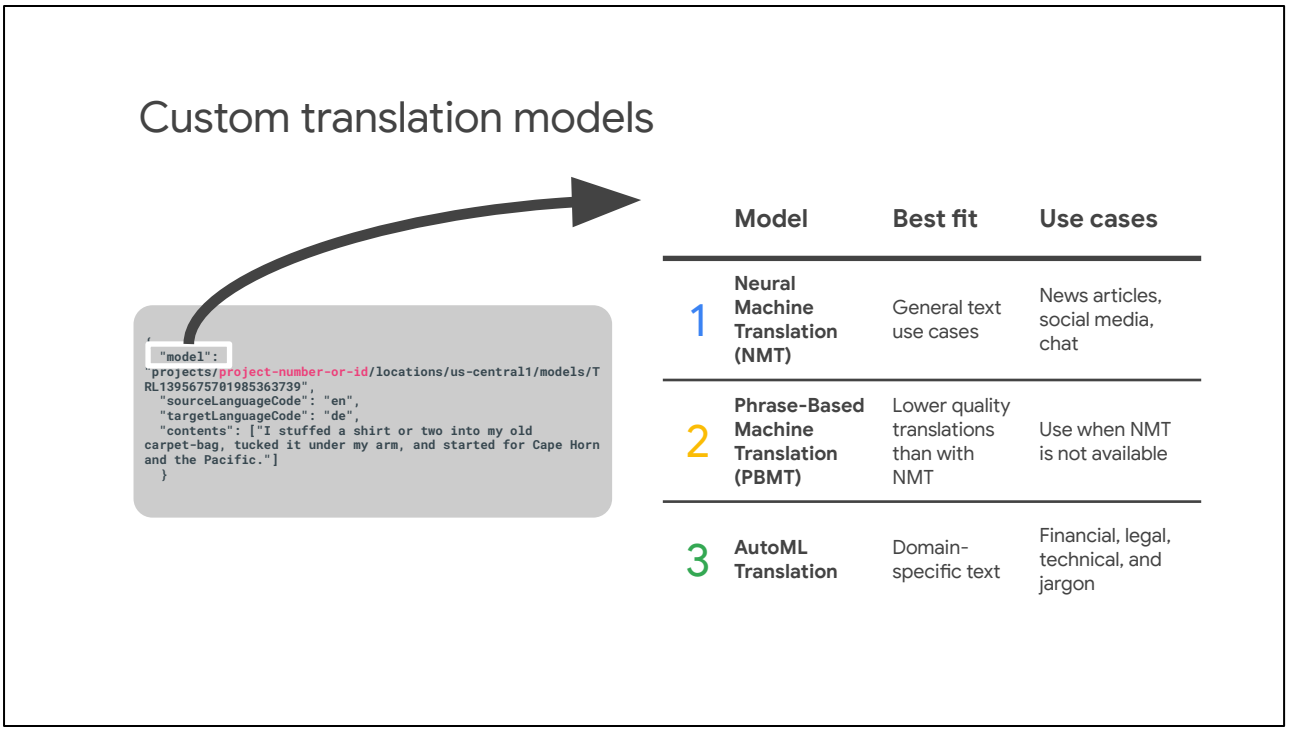
[Slide4]

Let’s see how to make a simple Translation API request. To translate text, you make a POST request and provide JSON in the request body to identify the target language and the text to translate. The input text can be plain text or HTML, and the output retains any HTML tags untranslated.



[Slide5]

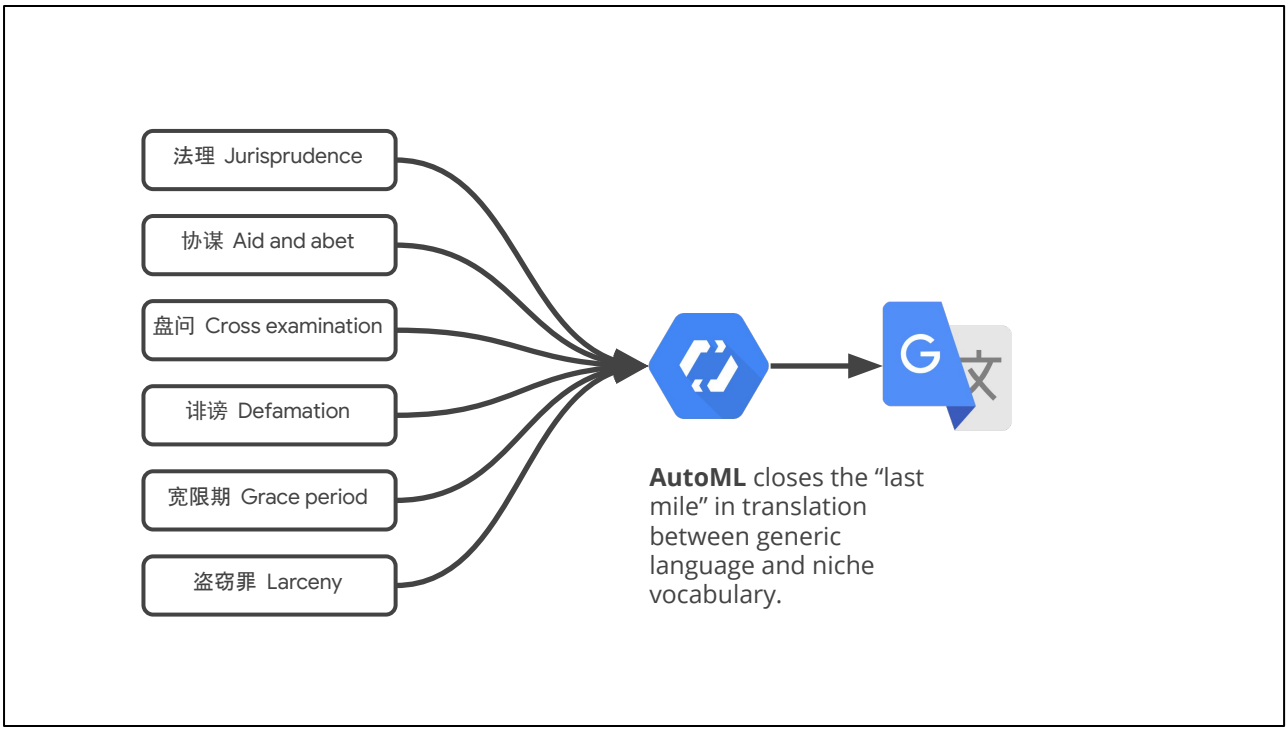
Sometimes you may need to detect the input language. This is easily done with a DETECT request. Save the input string in a file request.json and POST it like this. The response returns the language with a stated degree of confidence.



[Slide5]
The Advanced Edition supports custom translation models and glossaries, making it perfect for domain-specific translation tasks like financial services or legal.

When you send a translation request using the Advanced Edition, you can specify which type of model to use. There are three:

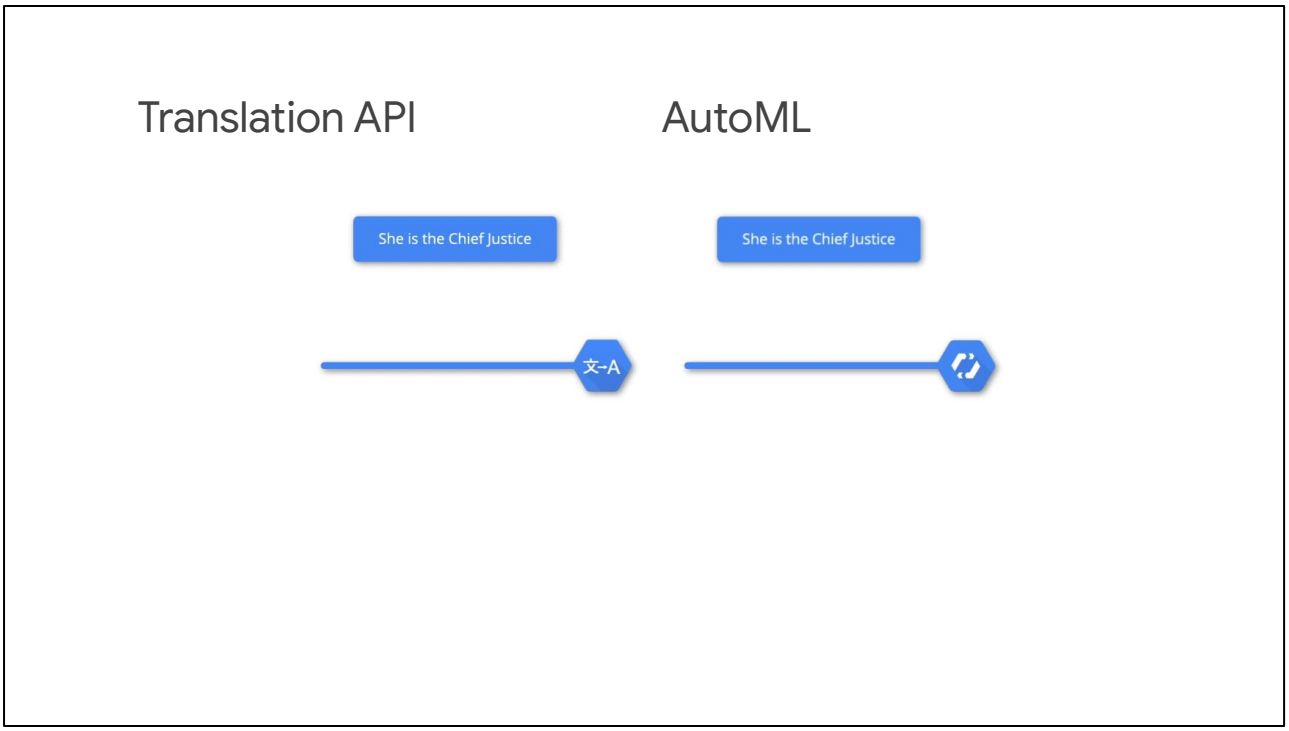
Neural Machine Translation: NMT is for general text use cases such as common website content that is not specific to a domain, such as news articles.
Phrase-Based Machine Translation: PBMT is the translation of sequences of words in blocks or ‘phrases.’ It might be used by default if the NMT model doesn't exist for a language pair.
Generally, NMT results in better-quality translations. And finally,
AutoML Translation: This is for domain-specific text. Customers provide training data specific to their use cases. These domain-specific phrases are used to customize Google's base model.



[Slide6]

The Translation API does a great job with general-purpose text.

Where AutoML Translation performs best is for the "last mile" between generic translation tasks and specific niche vocabularies. Custom models start from the generic Translation API model, but AutoML lets you add a layer that specifically helps the model get the right translation for domain-specific content. In order to train a custom model with AutoML Translation, you supply matching pairs of sentences in the source and target languages; that is, pairs of sentences that mean the same thing in the source and target languages.



[Slide7]

The closer in meaning your sentence pairs are, the better your model will work.

Media Translation API (beta)

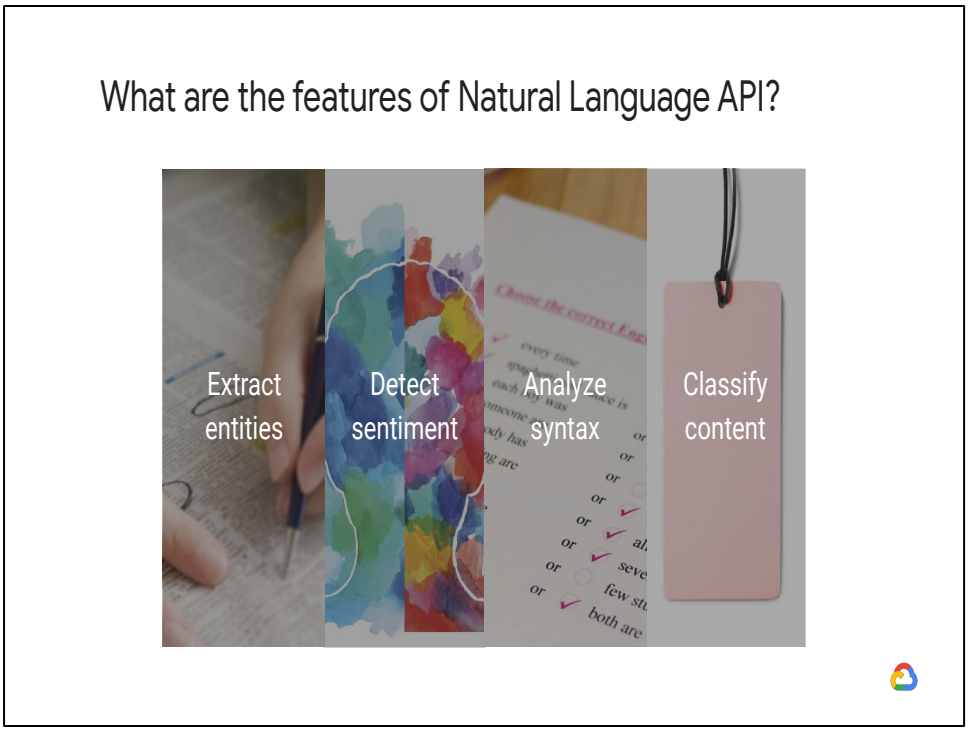
1. Uses features of both the Translation API and the Speech-to-Text API.
2. Handles multiple API calls to deliver seamless content translation.
3. Streams low latency, speedy translations from microphones or pre-recorded audio files.

[Slide8]

Media Translation API delivers real-time audio translation directly to your content and applications. It uses features of both the Translation API and Speech-to-Text API, taking care of multiple API calls to deliver seamless content translation of audio from live microphones or pre-recorded audio.

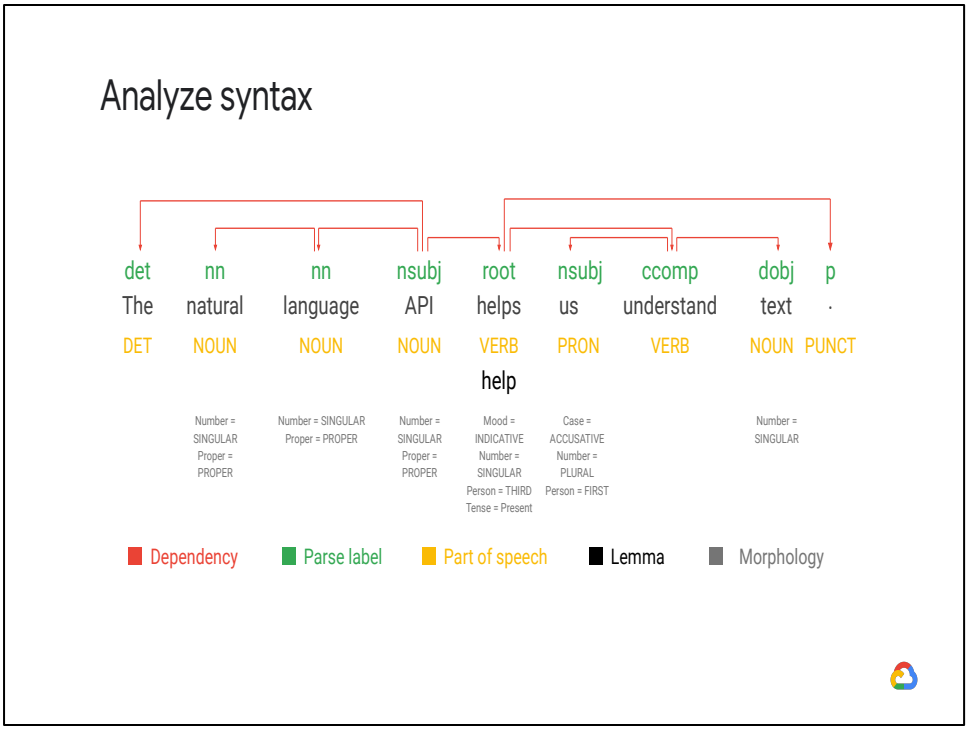
[Slide10]

Let’s see how Bloomberg use Google Translate API.

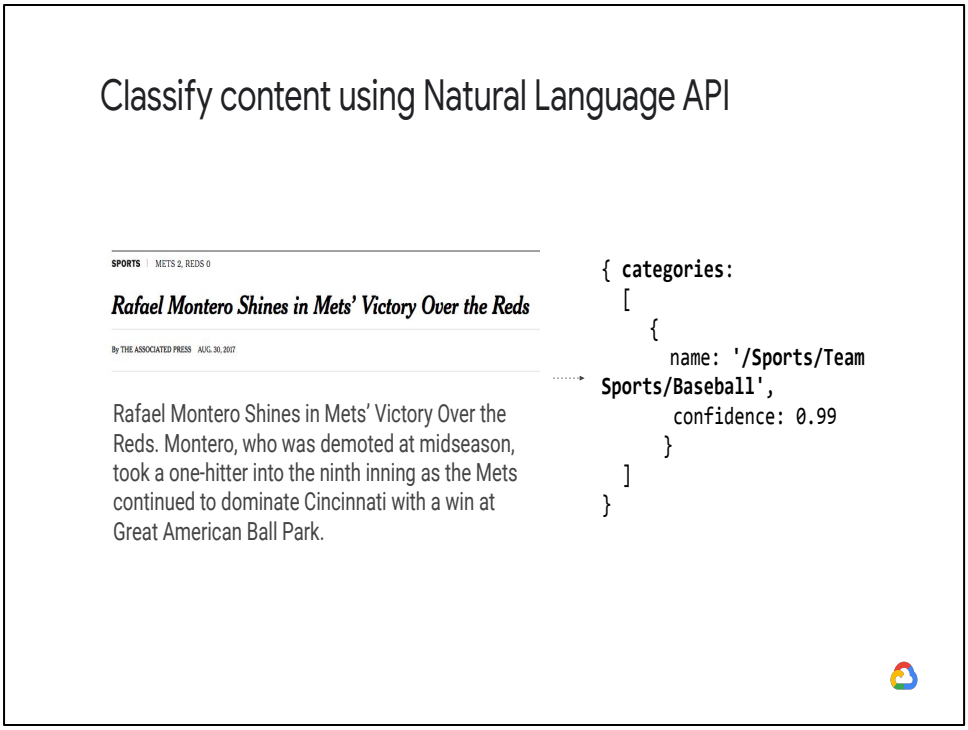


Here's everything you can do with the NL API:

- Entities: find key entities in your text
- Sentiment: is the text positive or negative?
- Syntax: extract linguistic details from your text, like subjects, verbs, and adjectives
- Classification: classify your text content into 1-2 of 700+ content categories (talk about media example, classifying articles)



Let’s dive into each of these features in a bit more detail, starting with syntax analysis



One of the API's newest features is text classification. The API has a database of more than 700 content categories; this feature will classify your text into one or more of these categories and provide a confidence score. This feature has a lot of possible applications in the media industry. Instead of manually classifying and tagging article content, you can do this easily with a single API call.

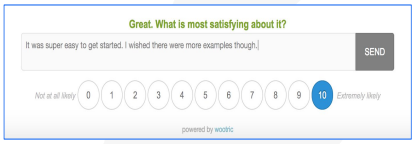
Wootric: Natural Language API in production

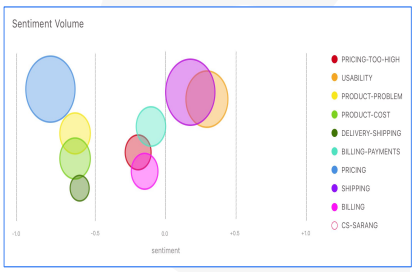
WOOTRIC

WIN CUSTOMERS FOR LIFE

Analyzing and routing feedback

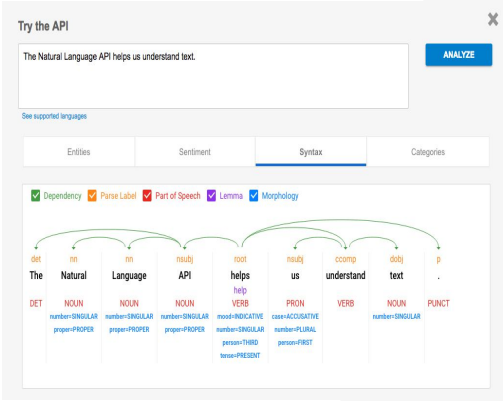
- Make sense of millions of qualitative customer feedback each week using **entity** and **sentiment analysis**.
- Route and respond to feedback in near real time, compared to manually classifying each response.





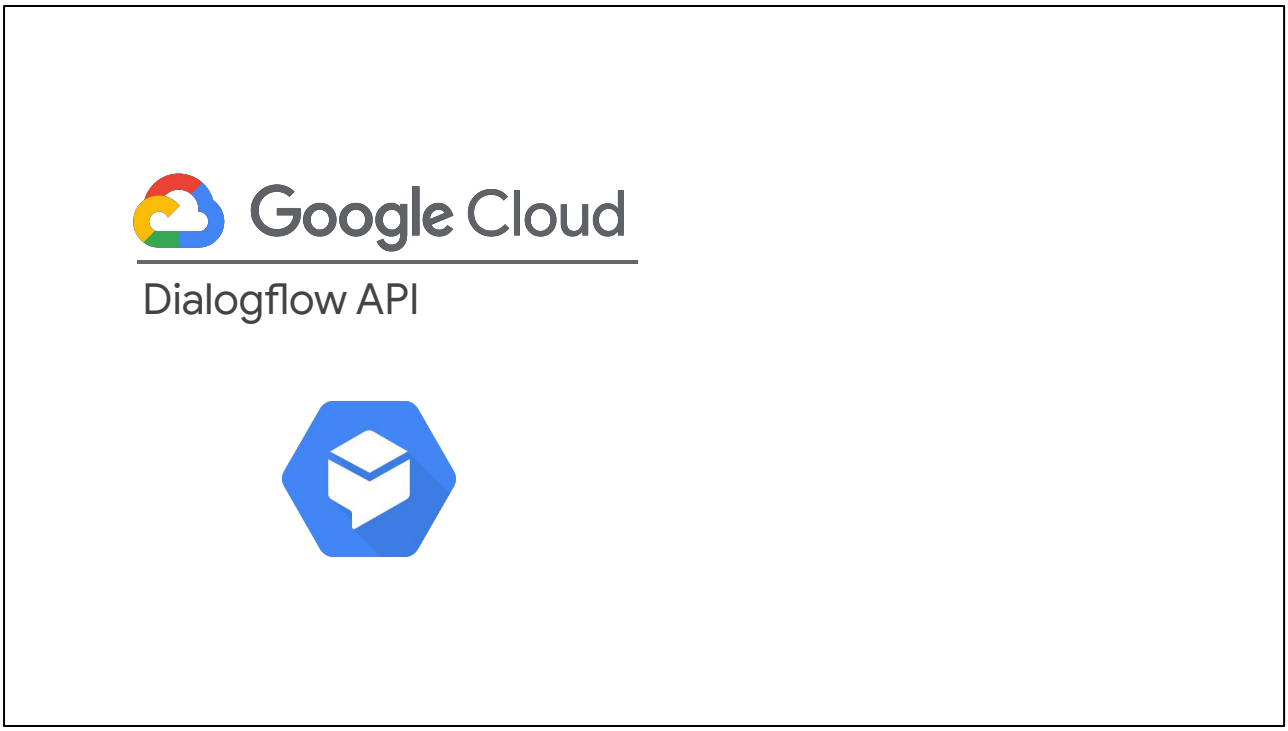
You can also try the NL API in the browser with your own text to see the resulting entities, sentiment, syntax and categories.

Try Natural Language API in the browser



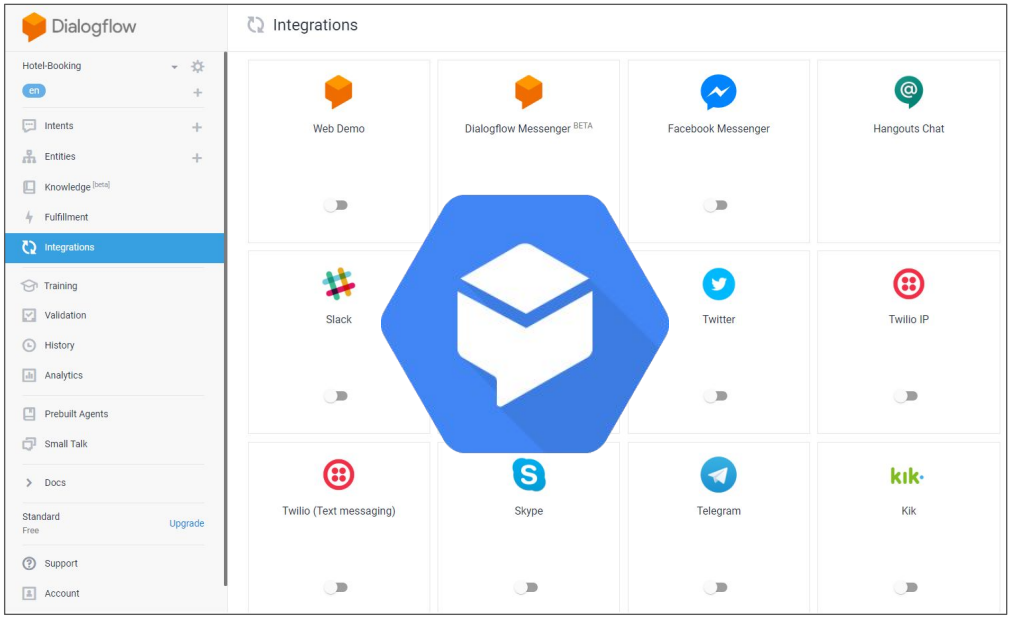
cloud.google.com/natural-language

You can also try the NL API in the browser with your own text to see the resulting entities, sentiment, syntax and categories.



[Slide 1]

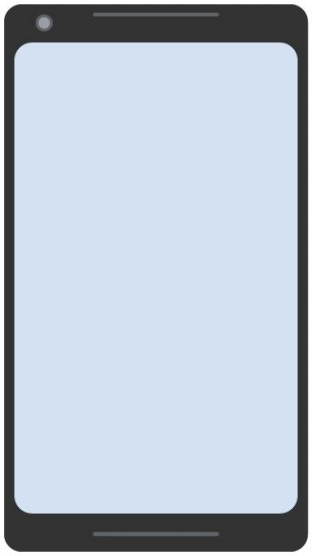
Dialogflow is used for building conversational interfaces. It can analyze text or audio and respond to a human in a natural, chatty way. Dialogflow is an authoring platform, not just an API, but it is possible to manage your chatbot through its API.



[Slide 2]

In most cases you would create a chatbot in the Dialogflow console and embed it into your telephony or text application using the pre-built integrations. But there are use cases when you might use the API instead. This might be for app development or in a device.

Before using the API, you should understand some basics of how Dialogflow and chatbots work.



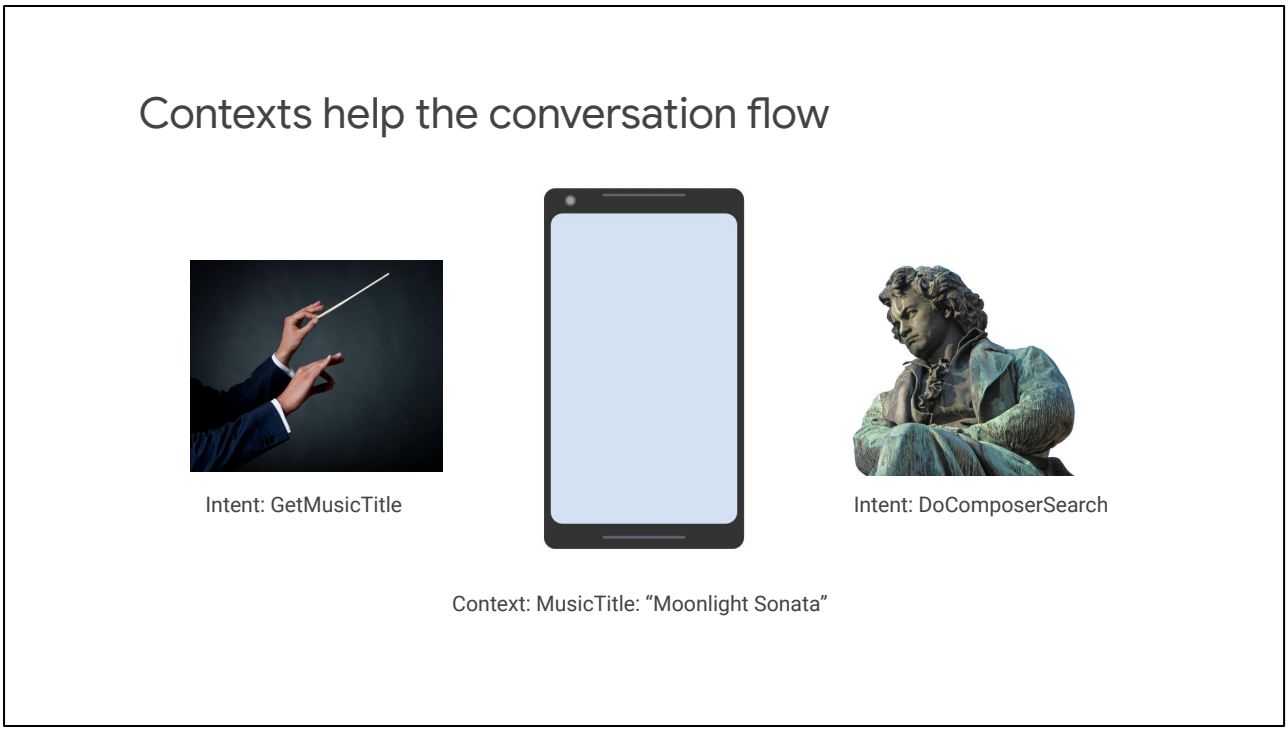
Intent	Example phrases	Response
Get opening times	What time do you open? Are you open yet? When do you open?	We open at six First sitting is six Not til six
Book a table	I'd like to book a table? Can I make a booking please Is it possible to book a table?	No problem Certainly Yes
Size the group	Do you have a table for <num>? We are a party of <num>? There are <num> of us?	Let me check One moment I'll check now

[Slide 3]

The building blocks of any chatbot are Intents and Entities.

Intents are the meaning behind something that the user types or speaks to a chatbot. So if I say “Are you open yet?” and the chatbot replies “We are open at six”, the Intent is “Get Opening Times.” Of course there are many different ways you might ask that question, so a chatbot has to be trained by giving it example phrases. You don’t have to think of every possible way of framing the question. Given enough samples, the ML algorithm gets very good at working out the Intent.

You also need to feed the chatbot its response. It needs to be told what to say once it understands the Intent. Just one sample response is enough, but having several creates more natural conversations.



[Slide 4]

A Dialogflow conversation takes place in a Session. You create the session at the beginning of the conversation and discontinue it when it has ended. Session data is only retained for 20 minutes, then deleted.

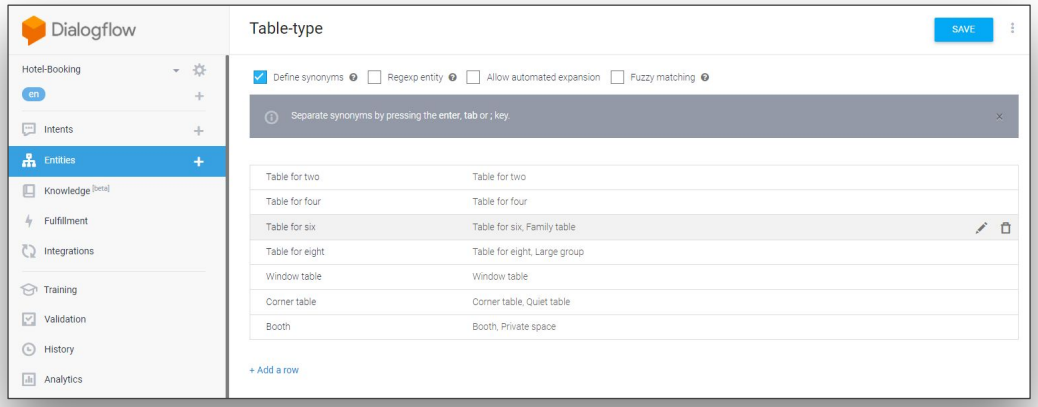
As with a human conversation, a Dialogflow chat works in turns: the user types something, Dialogflow detects its Intent and returns a response, which in turn may generate another user input.

Contexts help the conversation flow. They are used to pass parameter values from one Intent to the next. This is similar to how in a human conversation you infer what is meant by what was said previously.

When the user asks “Who wrote it?” a Context stores the value for the GetMusicTitle and passes it as a parameter to the next Intent, DoComposerSearch. This makes the conversation more natural.

Contexts last for the duration of a conversation session and then expire.

Entities are objects that your application acts on

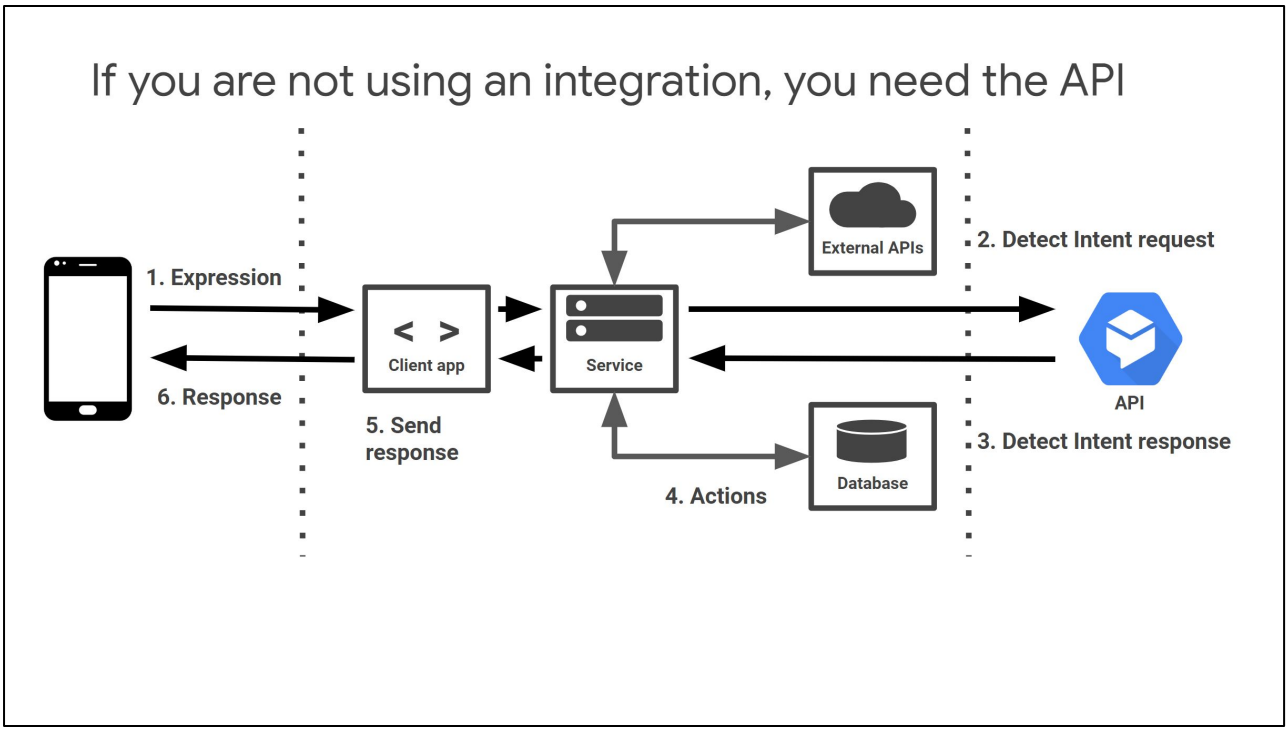


[Slide 5]

Entities are objects that your application acts on. The chatbot seeks to extract Entities from the Intents in the conversation and work out what to do with them. So if I type “Is it possible to book a table?”, the entities might be @bookings and @table-type. The actions that an entity initiates might be to ask the user another question, pass an input value to another application, or return an output value to the user. Entities have values, known as parameters.

You need to train Dialogflow to anticipate all the likely ways a human might reference an Entity parameter from within an Intent. For example, saying “I want a corner table” might reference an entity @table-type; but you also might reference the same entity by saying, “Can you get me a quiet table?” You have to train Dialogflow with all of the synonyms that reference the same entity.

Dialogflow is a complex and powerful authoring tool ,but if you understand Intents and Entities, and Contexts and Parameters, you have made a good start.



[Slide 6]

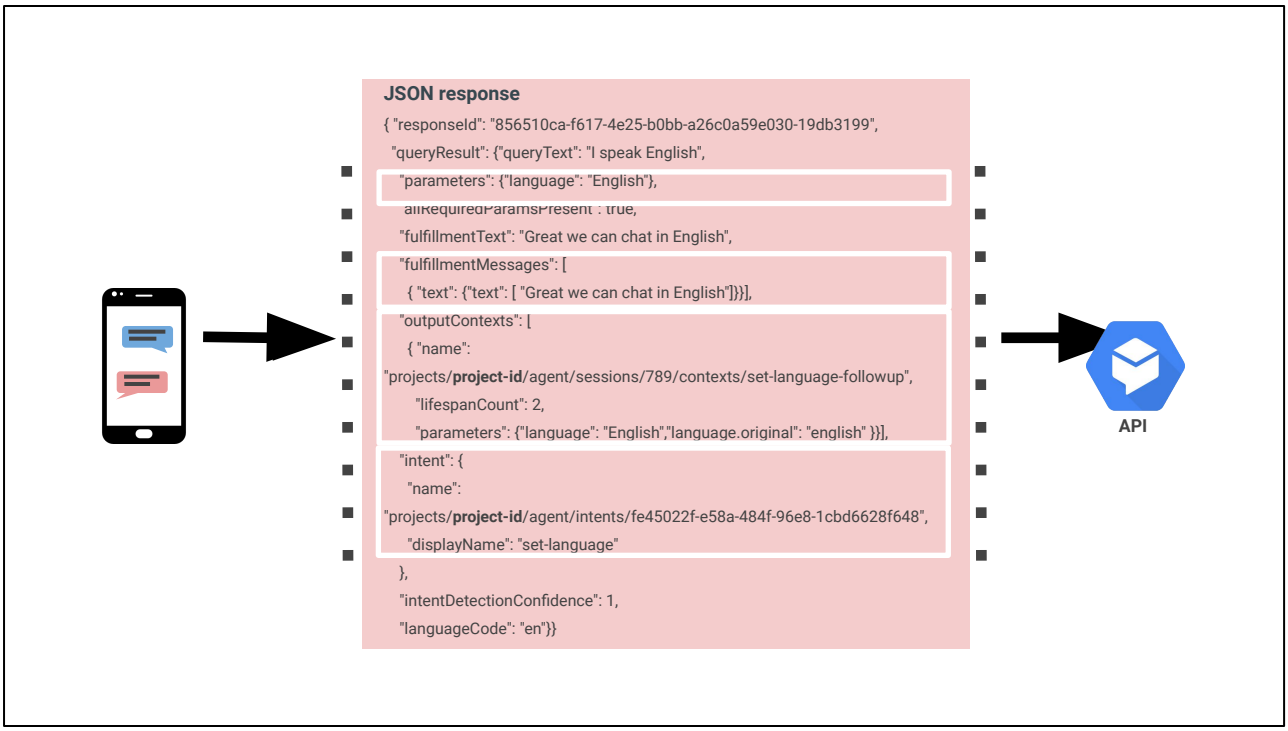
All of Dialogflow's functionality is accessible through the console, and the most common way to present the chatbot to the end user is through the pre-built telephony and text integrations with third party-applications, also configurable in the console. But sometimes you may want to access your chatbot in your own application through the API.

If you are not using an integration, you must write code to interact with end users. For each conversational turn, your code calls the Dialogflow API to query your agent.

You send the user's message to Dialogflow by calling the detectIntent or streamingDetectIntent methods of the Sessions type. detectIntent is used in conventional challenge/response type conversations; streamingDetectIntent is used for listening and responding concurrently. For example, with streamingDetectIntent, you could stream audio to Dialogflow and listen for a response at the same time. You could configure this further by having Dialogflow listen for pauses in the audio stream and interrupt the flow of the conversation. Just like a chatty human!

Whichever method you use, Dialogflow responds with information about the matched intent, the action, the parameters, and the response defined for the

intent. Your service performs actions as needed (for example, database queries or external API calls) and sends a message to the end user. This process continues until the conversation has ended.



[Slide 7]

Let's see how that works in practice. First you call the ***detectIntent*** method including your project-id and a session-id in the request.

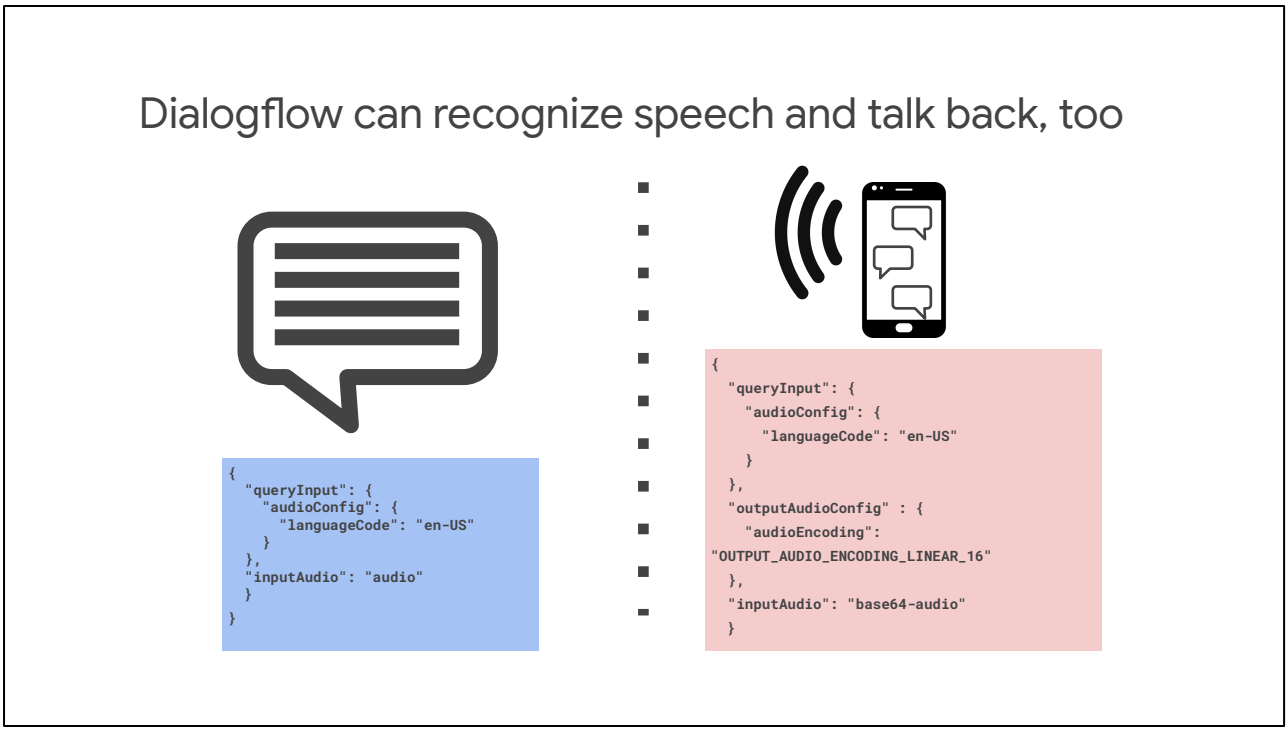
The accompanying JSON body is very simple. All that is needed in your JSON file is the query-input field with the text string and the language ID.

The JSON response is more complex.

In this example, note the following about the response:

- The **queryResult.intent** field contains the matched intent.
- The value of the **queryResult.fulfillmentMessages** field contains the intent response. This is the response that your system should forward to the end user.
- The value of the **queryResult.parameters** field contains the parameters extracted from the end-user expression.
- The **queryResult.outputContext** field contains the active context.

The *detectIntent* method and its JSON response is the core interaction you will make through the API.



[Slide 8]
A detectIntent request can be configured in powerful ways.

For example, it can be configured so that DialogFlow recognizes speech. You specify in the detectIntent that the input is audio, and Dialogflow processes the audio and converts it to text before attempting an intent match.

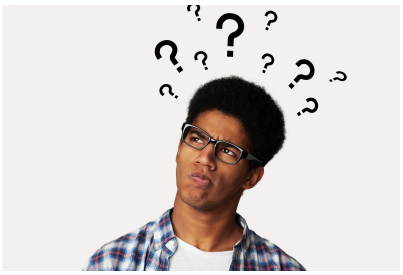
And if you need your bot to talk back to a user, it can do that too.

This is all done by setting parameters in the JSON body of your detectIntent request to the API.

A detectIntent can be configured in powerful ways

Stuffy nose

```
{
  "phrases": [
    Stuffy nose
  ],
  "boost": -3
}
```



Stuff he knows

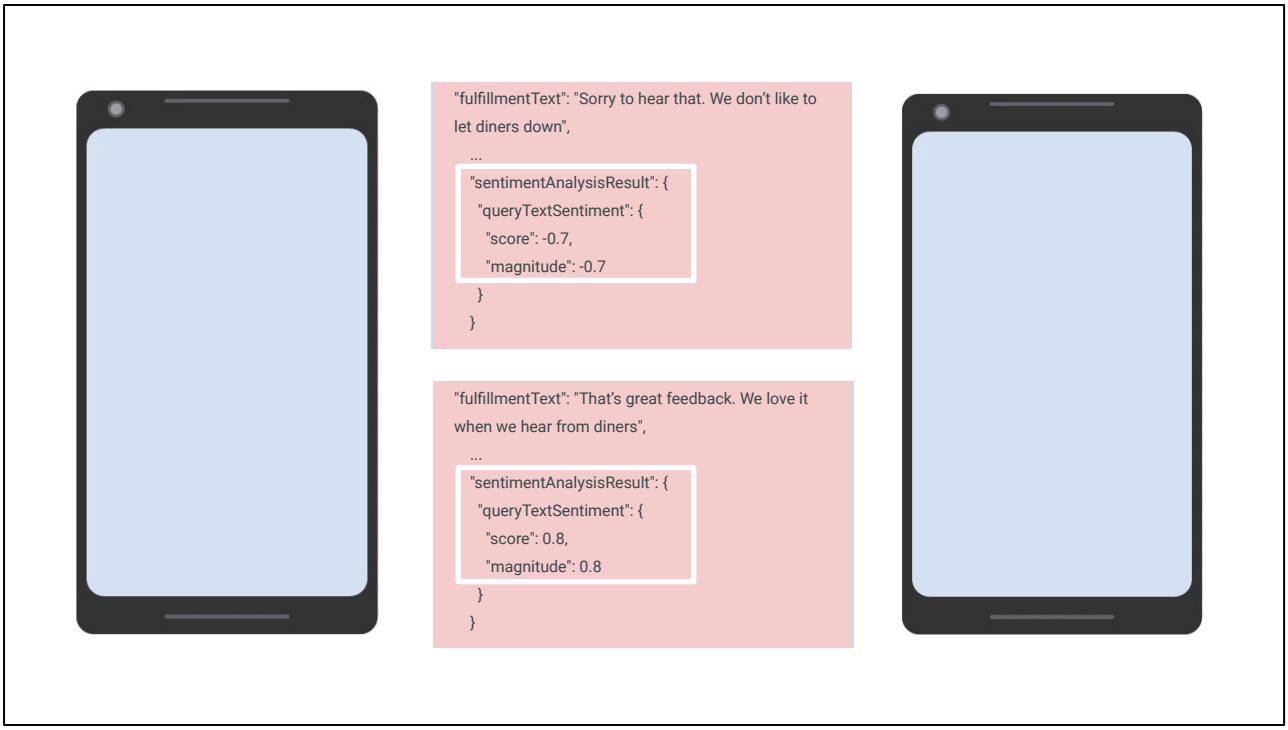
```
{
  "phrases": [
    Stuff he knows
  ],
  "boost": 15
}
```

[Slide 9]
By default Dialogflow is configured with Auto Speech Adaptation on, so that your agent will use entities, training phrases, and conversation state as hints to the meaning.

But you can strength this further in your detectIntent request by providing SpeechContext parameters as phrases with boost values, to prefer one meaning over another.

If the boost is positive, Dialogflow will increase the probability that the phrases in this context are recognized over similar-sounding phrases.

This might be useful in a contact center use case, where there tends to be highly repetitive and domain-specific use of language.



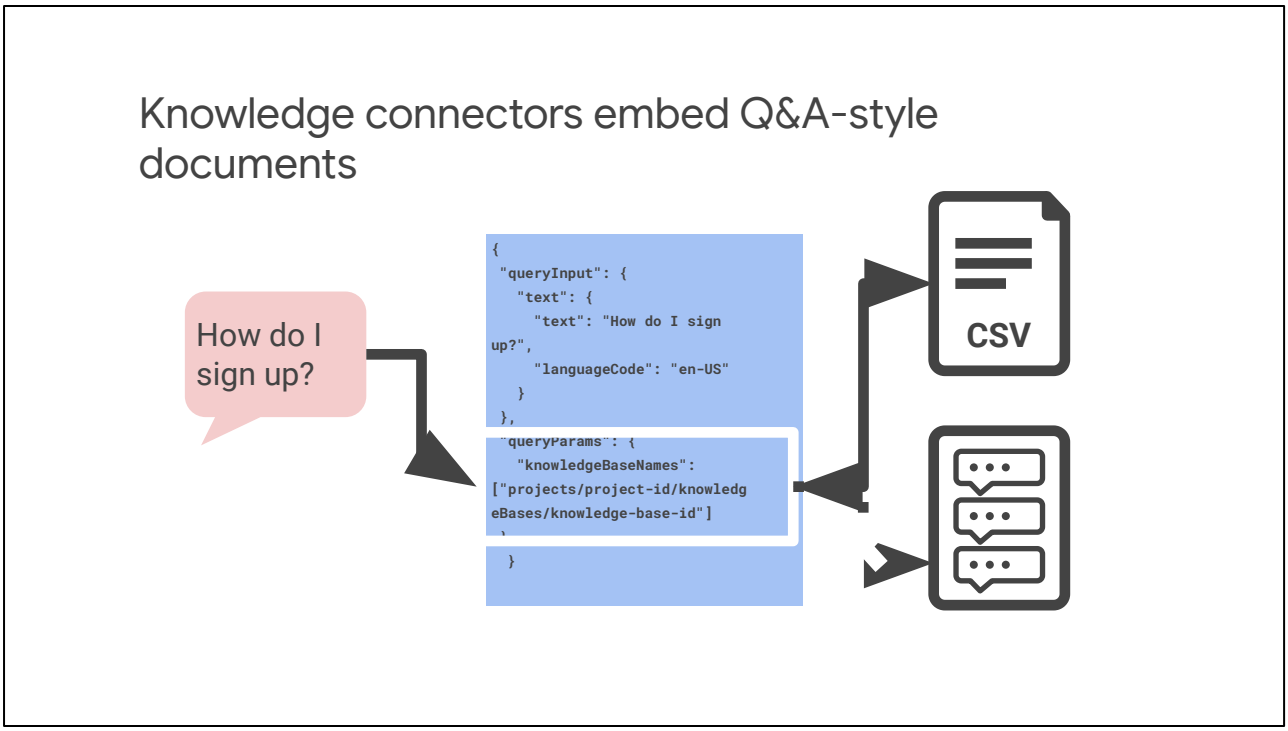
[Slide 10]

You can even configure sentiment analysis to detect whether a user is expressing a positive, negative, or neutral opinion.

To set this up, you configure the parameter analyzeQuerySentiment to true, and in the response you get back a positive, negative, or zero sentiment score.

This enables Dialogflow to return a more targeted message in the JSON response.

You can trigger sentiment analysis per detect intent request, or you can configure your agent to always return sentiment analysis results.



[Slide 11]

In some cases, for example with web-based FAQs, where you might already have all questions and answers in pre-existing documents, re-entering them into Dialogflow may not make sense. In this case you can configure Dialogflow to parse your documents—html and csv— for a suitable response, using knowledge connectors.

With knowledge connectors enabled, all detectIntent requests will try to find automated responses in your knowledge bases. Alternatively, you can specify one or more knowledgebases in individual detectIntent requests. To do this using the API, you would specify the knowledgebase as a query parameter in your input JSON package.

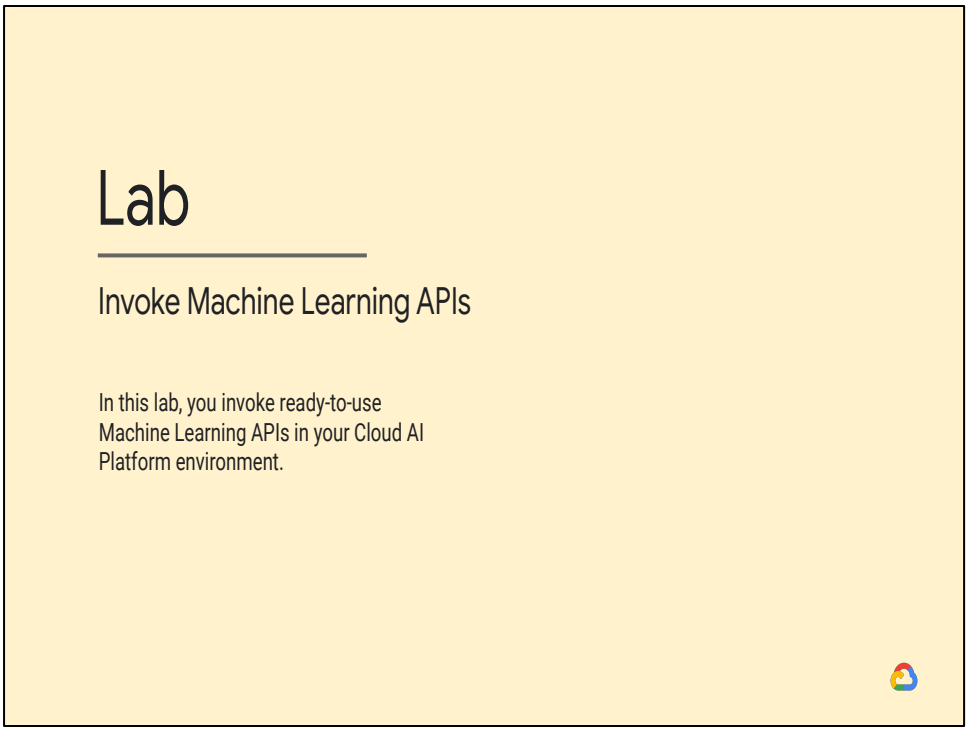
Knowledge connectors offer less response precision and control than intents, so the user agent conversation may be stilted. It is common for an agent using knowledge connectors to also use defined intents. In general, use intents to handle complex user requests, and let knowledge connectors handle simple requests.

Otto combines Intents with a knowledge base through the API

[Slide 12]
Otto is a Dialogflow API-driven chatbot developed by LearningPool for a European travel operator. It combines open-ended questioning of the chatbot with structured questions to a knowledgebase to inform staff about corporate brand, culture, and values.

https://tuiotto.learningpool.com

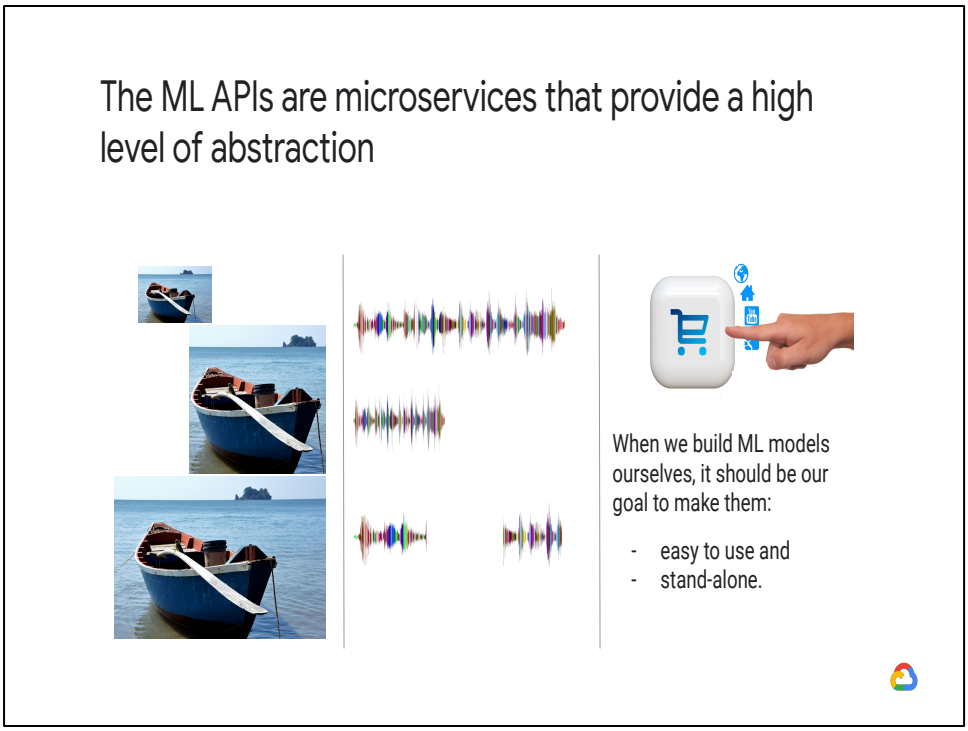
(Approval given by Henrietta Palmer, Learning Solutions Manager at TUI.)



In this lab, you will clone the GitHub repository of the course within your Cloud AI Platform instance and save it all in your project.
Then, you can work with a notebook to try out the ML APIs.

Link to Lab:
Invoking Machine Learning APIs (Qwiklabs link) Git repo link here.

As before, the labs often have you simply run through a worked-out example.
Because we cannot help you personally debug your code, we are choosing to do it this way.
To make sure you understand what is being done, though, try changing the code and see if you understand the impact of those changes.
Alternatively, remove the code from a critical section of the notebook (the critical sections are often pointed out in the "challenge" sections of the labs) and see if you can write the code yourself.
If you get stuck, look back at the solution.



Something to notice is how easy to use the ML APIs are. Sometimes, it is helpful to think about the alternatives to realize all the things the APIs abstract away.

For example,
Do the users of the Vision API need to send images of exactly the size that was used to *train* the Vision API? The NN must have been trained on fixed-size input images, but the resizing is handled for us.
Do the users of the Speech API need to time-window their audio samples? The NN must have been trained on small audio sequences, but we aren't expected to speed up or chop up our audio to fit the training data.

This idea of making ML predictions be microservices is another aspect of how Google does ML.
A Google search, for example, may end up in the invocation of 100+ microservices to return a result.

This, too, is something that you will learn how to do in this specialization; Google has an entire course—"Serving ML Predictions"—on this topic.
This is quite essential when it comes to putting ML models into production.

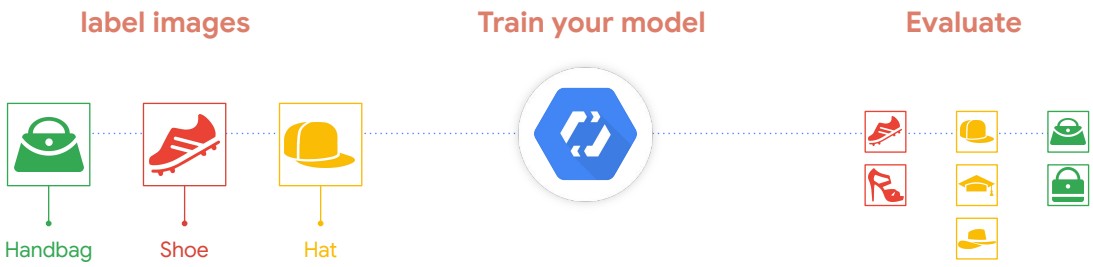
https://pixabay.com/en/sailboat-ship-sailing-greenland-459794/
https://pixabay.com/en/boat-fishing-sea-fishing-boat-oar-2933791/
https://pixabay.com/en/msc-cruises-venezia-port-boat-2940643/
https://pixabay.com/en/sound-wave-waveform-aural-audio-1781570/

https://pixabay.com/en/audio-aural-ear-hearing-music-2028515/
https://pixabay.com/en/sound-wave-waveform-aural-audio-1781569/
https://pixabay.com/en/hand-thumb-sign-ok-yes-positive-311121/
https://pixabay.com/en/social-media-icon-hand-keep-2489594/
https://pixabay.com/en/shopping-cart-icon-hand-finger-1276260/

Agenda

AutoML

ML Problem: Train a model to classify images



Google Cloud

Interior slide with large text over white

Introducing Cloud AutoML

An Algorithm that can create a Machine Learning Model

Cloud AutoML

allowing your developers to create high quality custom Models

Production Ready: Use it immediately, Cloud AutoML is already deployed in GCP, Scales well with GCP large scale computation resource

Human Data Labeling: For customers with no labeled training images, our in-house human labelers are available to review your custom instructions and label your images accordingly for model training.

Google Cloud

Title slides to top. Auto ML machine animates on screen

Why Cloud AutoML?

Limited ML expertise needed

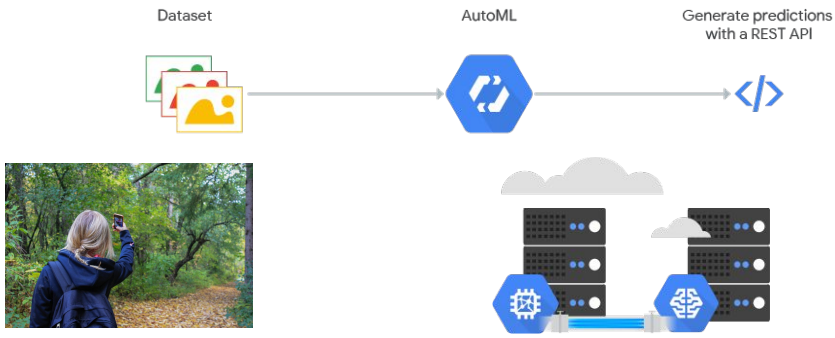
Your own custom models

Simple

High quality

Title slides to top. Auto ML machine animates on screen

How does this translate to building an ML model to identify diseased leaves?

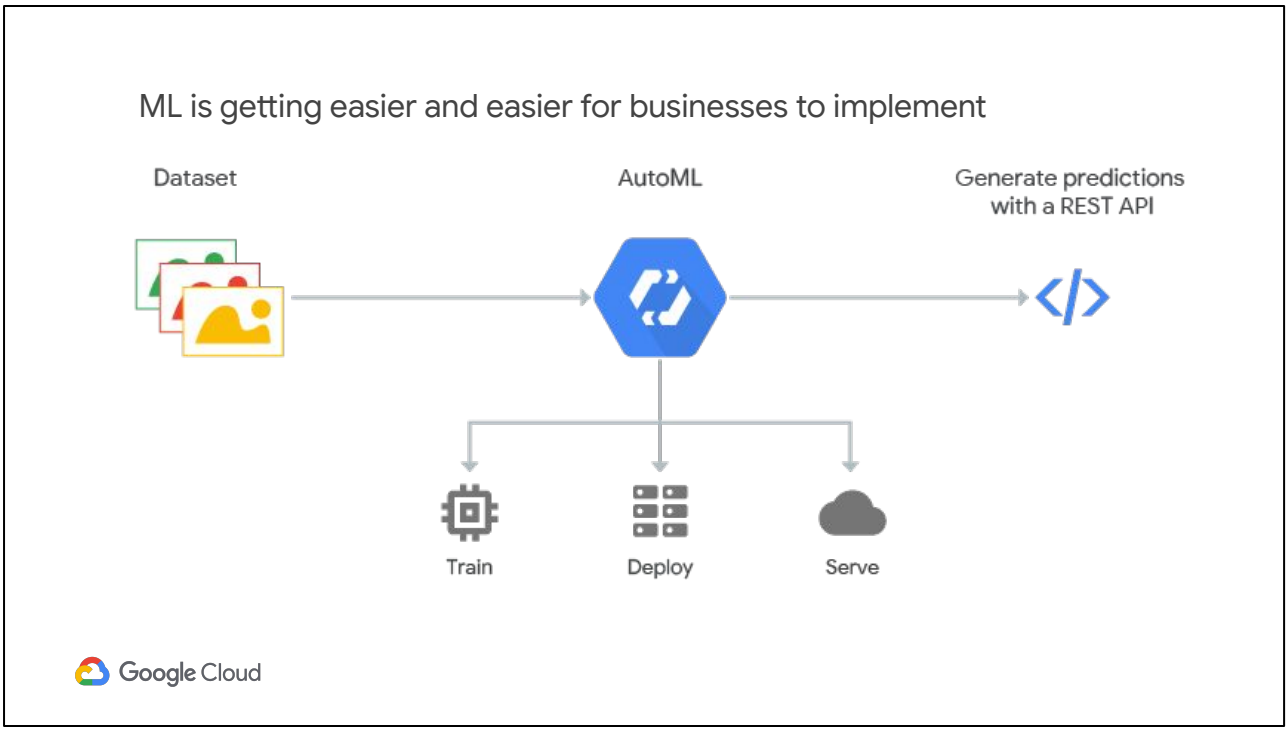


Google Cloud

Let me show you an example. Imagine we want to build that ML model to identify diseased leaves. Remember, we can do that using a standard algorithm for image classification. You don't need a PhD in image processing.

But back to our ML model. Another critical ingredient for ML is data. We need to collect lots of images of leaves. Today, you can do that easily with the camera on your phone.

Finally, we need the hardware and software to make that happen. That's now easier than ever before. We can use the cloud to power our ML model so that we can do it cost effectively. Next, we'll discuss the details of why that's possible today.



In this lab, you use the AutoML tool to train a custom model without writing any code.

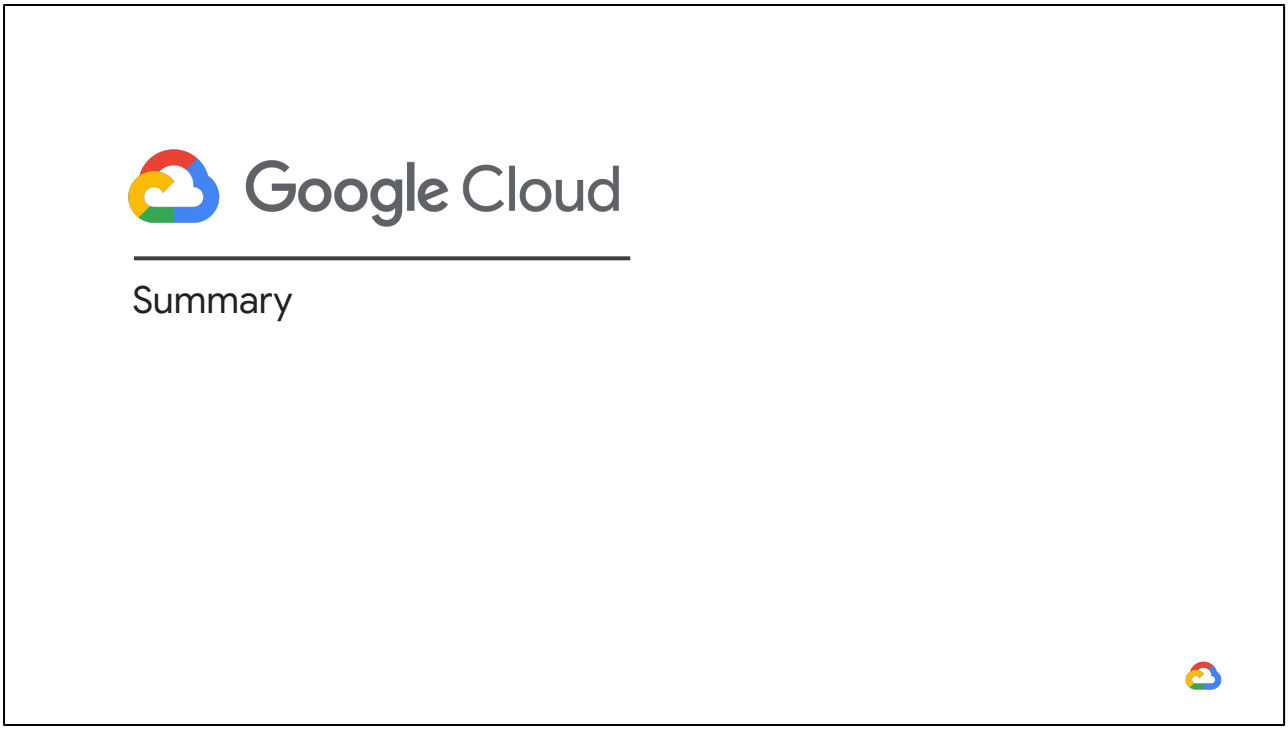
https://github.com/GoogleCloudPlatform/training-data-analyst/blob/master/courses/data-to-insights/demos/auto-ml-vision.md
(there is no sound; add your own narrative or follow the written narrative above)

Understanding Images with Google Cloud Platform
What the Vision API is
What tasks it can be used for
Its advantages and limitations
What AutoML vision is
How AutoML is different from the Vision API
How to use AutoML vision to train a custom vision model

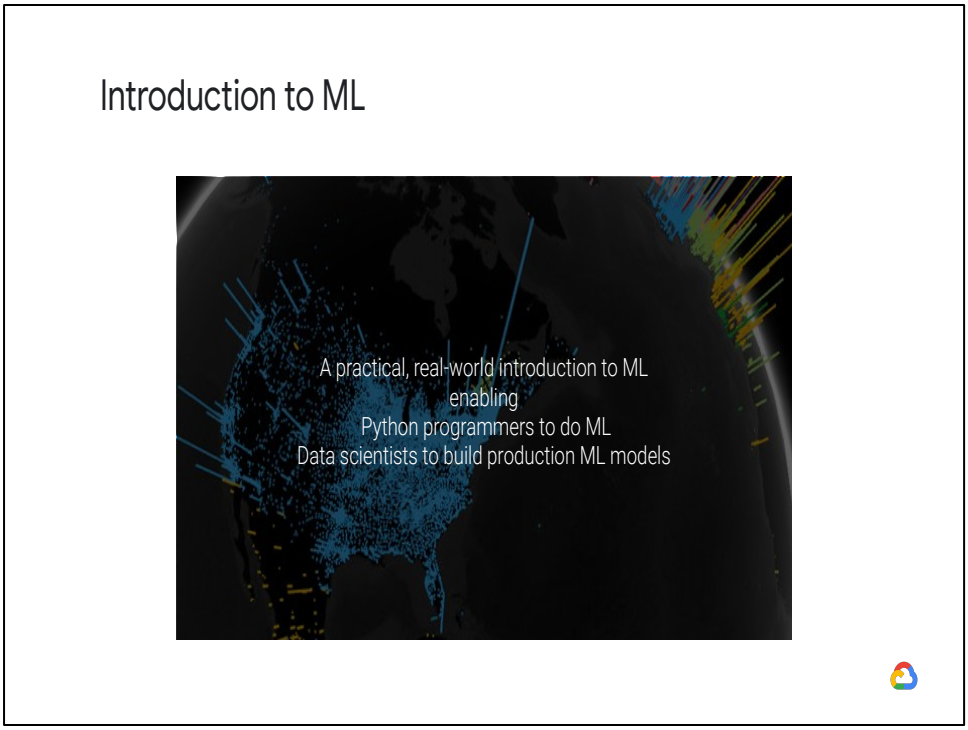
AutoML Vision vs Vision API

Attribute	AutoML Vision	Vision API
Objective	Enabling **developers with no ML expertise** to build state of the art ML models for Images	Enabling **ML practitioners** to harness power of Google's ML for Images
Primary use case	Classification	Face detection, OCR, Object detection etc.
Data requirements	Images with labelled data	Just Images (may or may not required labelled data)
Output format	Labels with probability	As per the problem
Custom requirements	Can't be customized	Can be used for any custom made solutions
Efforts	Low for solution designing	High for end to end model development
Status	Alpha phase; Limited processing	Publically available

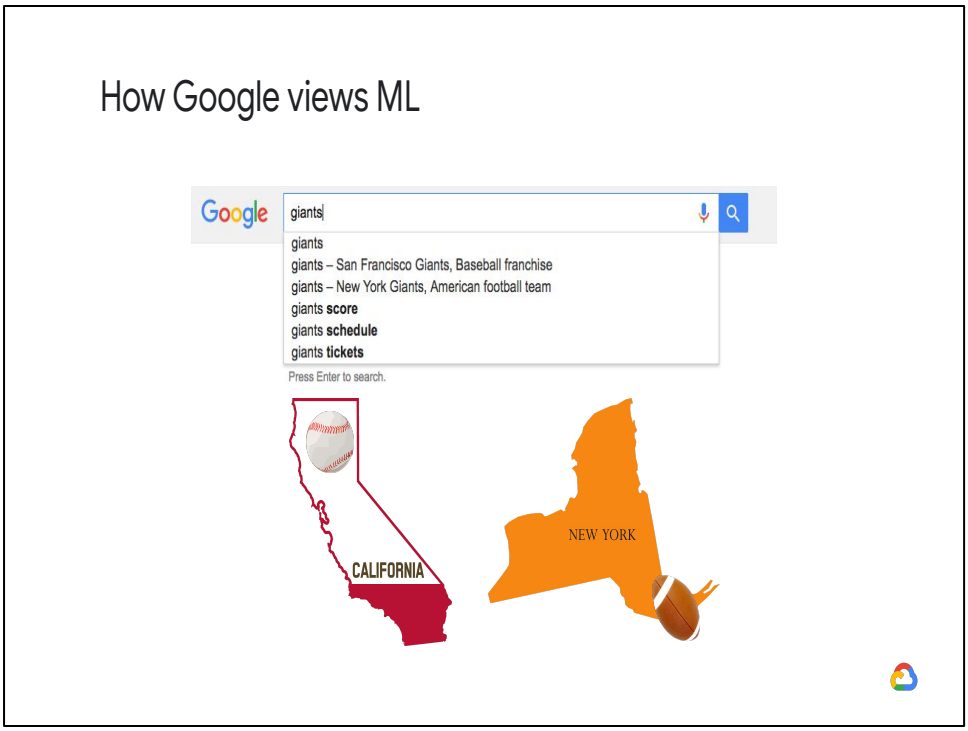
Interior slide with large text over white



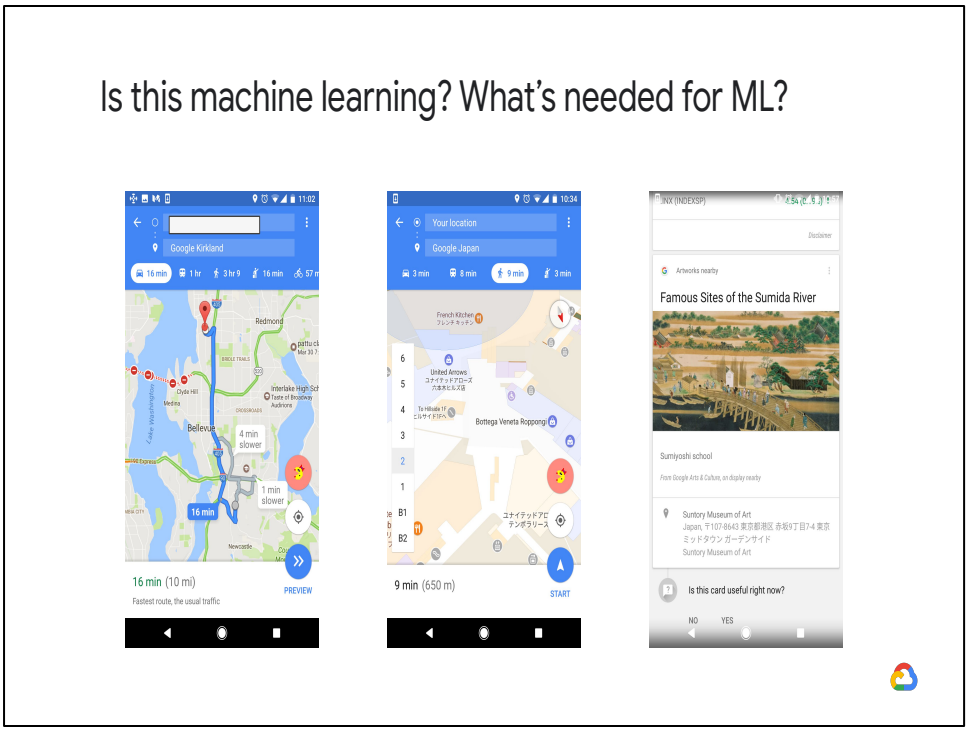
Welcome to the summary of this ‘How Google does Machine Learning’ chapter.



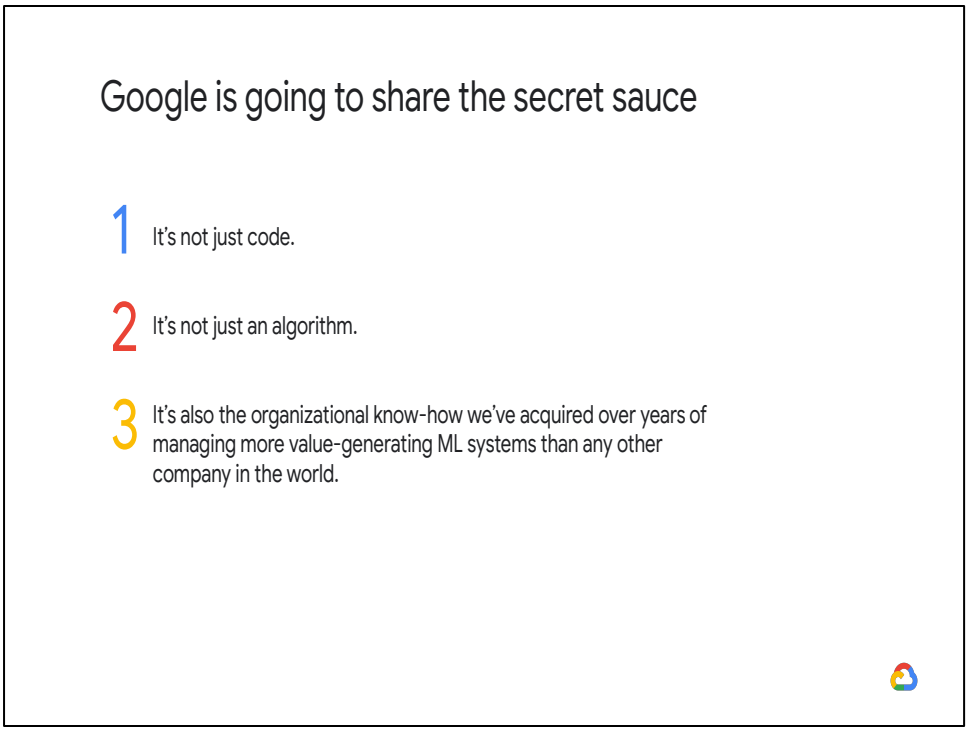
The aim of the specialization was to teach you how to build production machine learning models, whether you are a Python programmer or a data scientist. The specialization consists of a series of courses that provide a practical, real-world introduction to machine learning.



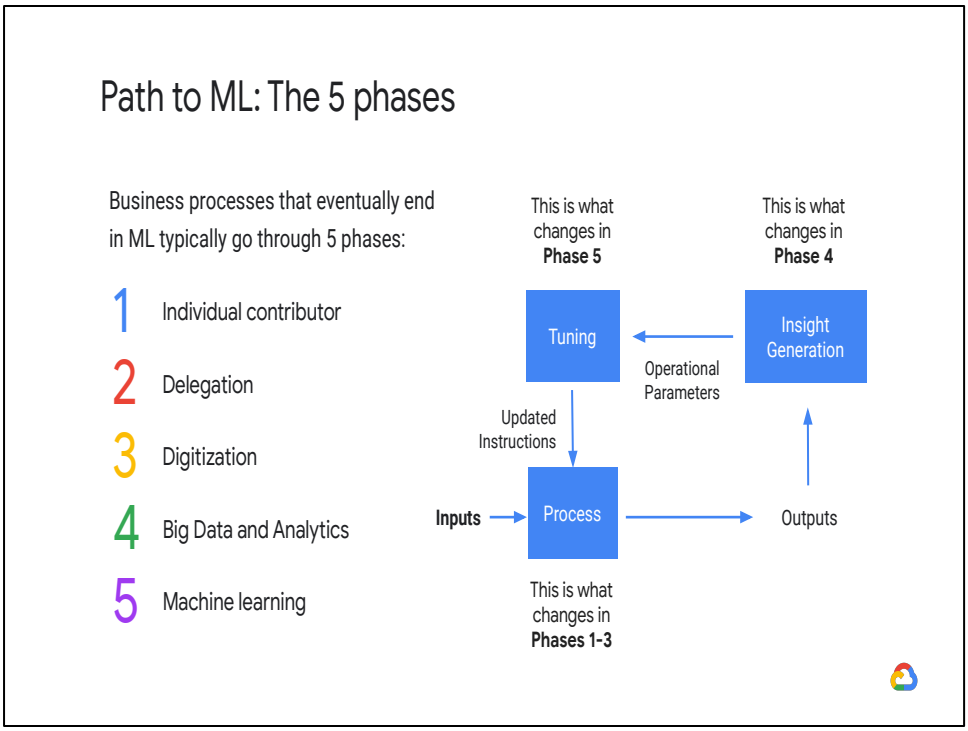
Then, we looked at how Google views ML, as a way to replace heuristic rules that build up over time.
For example, Google Search used heuristic rules to decide whether we needed to return a page for San Francisco Giants or New York Giants. This is now done using machine learning.



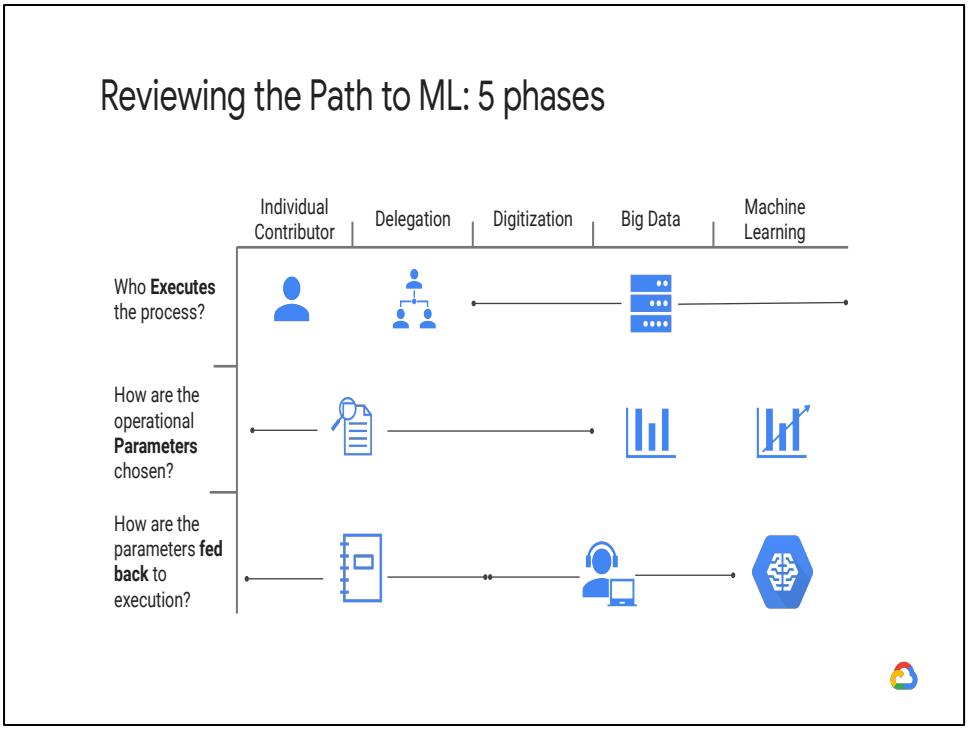
You then heard about ML as a way to increasingly personalize your business's offerings for your customers, using Google Maps as an example. For example, while routing directions between point A and point B can be done with deterministic algorithms, inferring which floor of a subway station you are in needs some amount of machine learning. Providing personalized recommendations is impossible to do at scale without ML.



You delved into the secret sauce behind ML and it turns out to be something quite unglamorous -- it's all about organizational know-how.



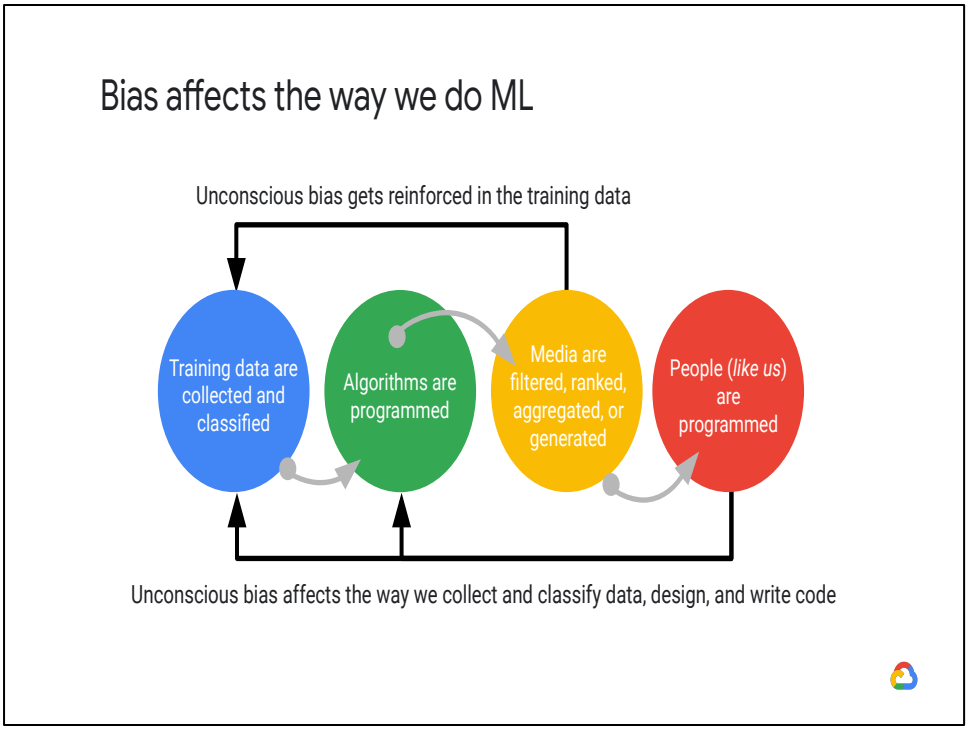
It is important to recognizing the five phases that business processes tend to go through and not skip any of these stages in order to be successful at ML.



It is also important to recognize how to transition between these phases thoughtfully

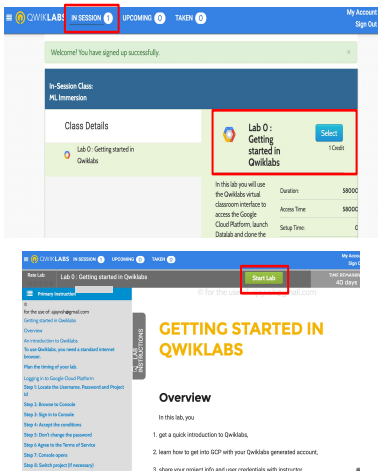
How to recognize biases that can be amplified because of the data you use

Finally, you talked about why fair is not the default in ML and how to recognize biases that can be amplified because of the data you use.



Unconscious bias affects the way you do machine learning and can get reinforced in the training data.
It affects the way you collect and classify data, design, and write code.

Use Qwiklabs to try out GCP

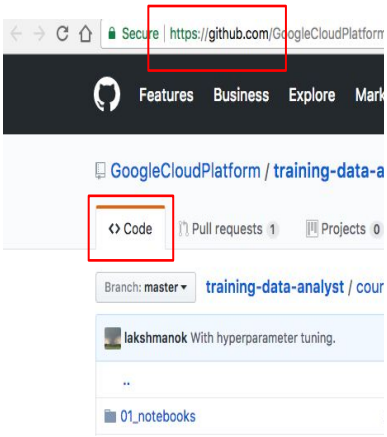


To give you a taste of ML, you were introduced to Qwiklabs as a way to quickly start trying out GCP.
This is a way for you to do the labs in this specialization without having to pay extra for the computing resources. Qwiklabs also has a number of other labs and quests -- you are strongly encouraged to practice and gain experience in machine learning and cloud technologies using Qwiklabs.

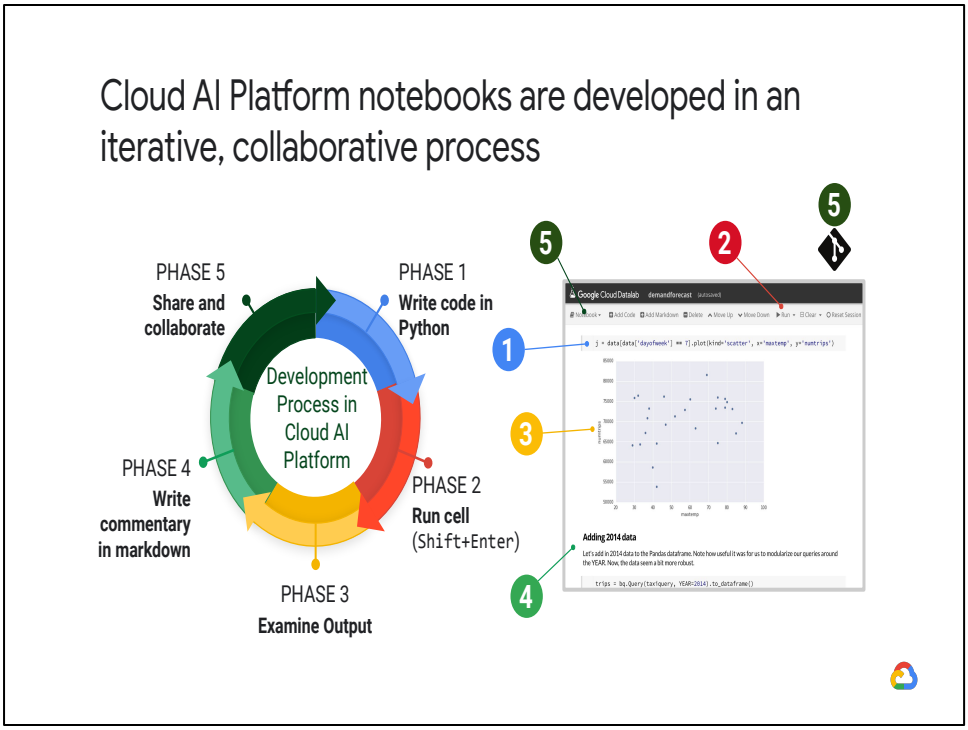
Source code for labs is on GitHub

https://github.com/GoogleCloudPlatform/training-data-analyst/tree/master/courses/machine_learning/deepdive

Later, practice taking the lab apart and trying to build it yourself on your own GCP account (strongly recommended).



You were told where to find the source code -- it's on github, it's open source and you should totally use our examples as a starting point for your projects.



You also looked at Cloud Cloud AI Platform. Python notebooks are the tool of choice for data scientists and is the way you will do the majority of hands-on activities in this specialization.

The notebooks are already worked out, and you are encouraged to pause and think through as you go through the notebooks step-by-step. If you are up to the challenge, you could also remove some of the key cells and see if you can write the necessary code yourself.

Notes:
https://pixabay.com/en/hard-drive-hdd-technology-digital-870699/ (cc0)

You heard about how Cloud Storage and Compute Engine provide the CPU and storage necessary for ephemeral, distributed notebooks.

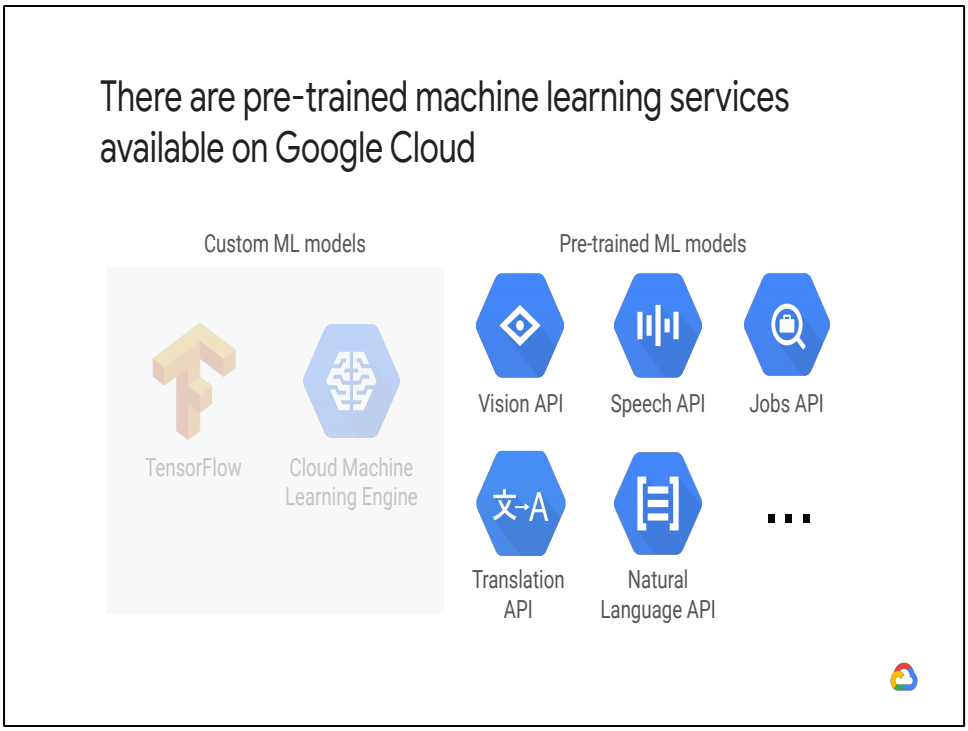
Demo: Query large datasets in seconds

```
#standardsql

# medicare claims in 2014
SELECT
  nppes_provider_state AS state,
  ROUND(SUM(total_claim_count) / 1e6) AS
total_claim_count_millions
FROM
  `bigquery-public-data.medicare.part_d_prescriber_2014`
GROUP BY
  state
ORDER BY
  total_claim_count_millions DESC
LIMIT 5;
```

Row	state	total_claim_count_millions
1	CA	116.0
2	FL	91.0
3	NY	80.0
4	TX	76.0
5	PA	63.0

And used the notebook to launch BigQuery queries, on thousands of machines at scale.
This, for example, is a query that is carried out on a dataset of millions of healthcare claims.

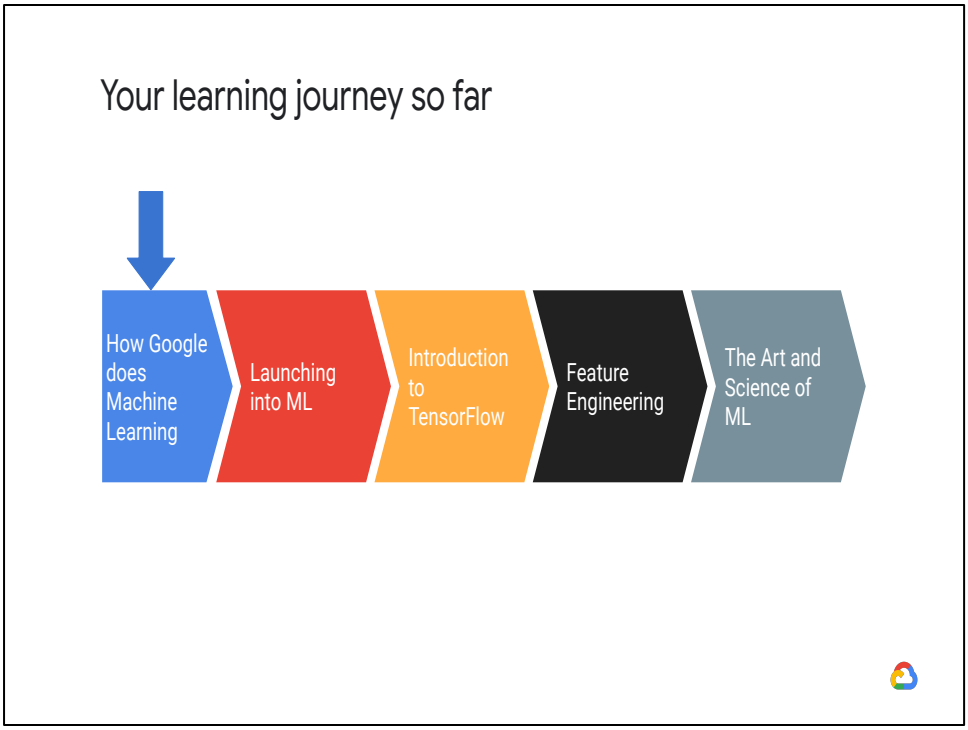


You also invoked pre-trained ML models that are available as APIs -- as machine learning matures, many of the reusable tasks will be available in pre-trained form.

The ML APIs are microservices that provide a high level of abstraction

When we build ML models ourselves, it should be our goal to make them as easy to use and stand-alone.

A key point that the ML APIs teach us is that we want take our ML models and make them just as easy to use.



And that brings us to this chapter: How Google Does Machine Learning