History of Google Big DataQuery:
The final layer of the Google Cloud infrastructure that is left to explore is big data and machine learning products. In this video, we'll examine the evolution of data processing frameworks through the lens of product development. Understanding the chronology of products can help address typical big data and ML challenges. Historically speaking, Google experienced challenges related to big data quite early– mostly with large datasets, fast-changing data, and varied data. This was the result of needing to index the World Wide Web And as the internet grew, Google needed to invent new data processing methods. So, in 2002, Google released the Google File System, or GFS. GFS was designed to handle data sharing and petabyte storage at scale. It served as the foundation for Cloud Storage and also what would become the managed storage functionality in BigQuery. A challenge that Google was facing around this time was how to index the exploding volume of content on the web. To solve this, in 2004 Google wrote a report that introduced MapReduce. MapReduce was a new style of data processing designed to manage large-scale data processing across big clusters of commodity servers. As Google continued to grow, new challenges arose, specifically with recording and retrieving millions of streaming user actions with high throughput. The solution was the release in 2005 of Cloud Bigtable, a high-performance NoSQL database service for large analytical and operational workloads. With MapReduce available, some developers were restricted by the need to write code to manage their infrastructure, which prevented them from focusing on application logic. As a result, from 2008 to 2010, Google started to move away from MapReduce as the solution to process and query large datasets. So, in 2008, Dremel was introduced. Dremel took a new approach to big-data processing by breaking the data into smaller chunks called shards, and then compressing them. Dremel then uses a query optimizer to share tasks between the many shards of data and the Google data centers, which processed queries and delivered results. The big innovation was that Dremel autoscaled to meet query demands. Dremel became the query engine behind BigQuery. Google continued innovating to solve big data and machine learning challenges. Some of the technology solutions released include: Colossus, in 2010, which is a cluster-level file system and successor to the Google File System. BigQuery, in 2010 as well, which is a fully-managed, serverless data warehouse that enables scalable analysis over petabytes of data. It is a Platform as a Service (PaaS) that supports querying using ANSI SQL. It also has built-in machine learning capabilities. BigQuery was announced in May 2010 and made generally available in November 2011. Spanner, in 2012, which is a globally available and scalable relational database. Pub/Sub, in 2015, which is a service used for streaming analytics and data integration pipelines to ingest and distribute data. And TensorFlow, also in 2015, which is a free and open source software library for machine learning and artificial intelligence. 2018 brought the release of the Tensor Processing Unit, or TPU, which you'll recall from earlier, and AutoML, as a suite of machine learning products. The list goes on till Vertex AI, a unified ML platform released in 2021. And it's thanks to these technologies that the big data and machine learning product line is now robust. This includes: Cloud Storage, Dataproc, Cloud Bigtable, BigQuery, Dataflow, Firestore, Pub/Sub, Looker, Cloud Spanner, AutoML, and Vertex AI, the unified platform. These products and services are made available through Google Cloud, and you'll get hands-on practice with some of them as part of this course.

As we explored in the last video, Google offers a range of big data and machine learning products. So, how do you know which is best for your business needs? Let's look closer at the list of products, which can be divided into four general categories along the data-to-AI workflow: ingestion and process, storage, analytics, and machine learning. Understanding these product categories can help narrow down your choice. The first category is ingestion and process, which include products that are used to digest both real-time and batch data. The list includes Pub/Sub Dataflow Dataproc Cloud Data Fusion You'll explore how Dataflow and Pub/Sub can ingest

streaming data later in this course. The second product category is data storage, and you'll recall from earlier that there are five storage products: Cloud Storage Cloud SQL Cloud Spanner Cloud Bigtable, and Firestore Cloud SQL and Cloud Spanner are relational databases, while Bigtable and Firestore are NoSQL databases. The third product category is analytics. The major analytics tool is BigQuery. BigQuery is a fully managed data warehouse that can be used to analyse data through SQL commands. In addition to BigQuery, you can analyse data and visualise results using: Looker, and Looker Studio. You will explore BigQuery, Looker, and Looker Studio in this course. And the final product category is machine learning, or ML. ML products include both the ML development platform and the AI solutions: The primary product of the ML development platform is Vertex AI, which includes the products and technologies: AutoML Vertex AI Workbench, and TensorFlow AI solutions are built on the ML development platform and include state-of-the-art products to meet both horizontal and vertical market needs. These include: Document AI Contact Center AI Retail Product Discovery, and Healthcare Data Engine These products unlock insights that only large amounts of data can provide. We'll explore the machine learning options and workflow together with these products in greater detail later.

With many big data and machine learning product options available. It can be helpful to see an example of how an organization has leveraged Google Cloud to meet their goals. In this video, you'll learn about a company called Gojek and how they were able to find success through Google Cloud's data engineering and machine learning offerings. The story starts in Jakarta, Indonesia. Traffic congestion is a fact of life for most Indonesian residents. To minimize delays, many rely heavily on motorcycles, including motorcycle taxis, known as Ojeks, to travel to and from work or personal engagements. Founded in 2010 and headquartered in Jakarta, a company called Gojeks started as a call center for Ojek bookings. The organization has leveraged demand for the service to become one of the few unicorns in Southeast Asia. A unicorn is a privately held startup business valued at over one billion US dollars. Since its inception, Gojek has collected data to understand customer behavior. In 2015, launched a mobile application that bundled ride-hailing, food delivery, and grocery shopping. They hit hyper growth very quickly. According to the Q2, 2021 Gojek fact sheet, the Gojek app has been downloaded over 190 million times. They have two million driver-partners and about 900,000 merchant partners. The business has relied heavily on the skills and expertise of the technology team and on selecting the right technologies to grow and to expand into new markets. Gojek chose to run its applications and data in Google Cloud. Gojek's goal is to match the right driver with the right request as quickly as possible. In the early days of the app, a driver would be pinged every 10 seconds, which meant six million pings per minute, which turned out to be eight billion pings per day across their driver-partners. They generated around five terabytes of data each day. Leveraging information from this data was vital to meeting their company goals. But Gojek faced challenges along the way. Let's explore two of them to see how Google Cloud was able to solve them. The first challenge was

data latency. When they wanted to scale their big data platform, they found that most reports were produced one day later, so they couldn't identify problems immediately. To help solve this, Gojek migrated their data pipelines to Google Cloud. The team started using Dataflow for Streaming Data Processing and BigQuery for real-time business insights. Another challenge was quickly determining which location had too many or too few drivers to meet demand. Gojek was able to use Dataflow to build a streaming event data pipeline. This let driver locations paying Pub/Sub every 30 seconds, in Dataflow would process the data. The pipeline would aggregate the supply pings from the drivers against the booking requests. This would connect to Gojeks notification system to alert drivers where they should go. This process required a system that was able to scale up to handle times of high-throughput and then back down again. Dataflow is able to automatically manage the number of workers processing the pipeline to meet demand. The Gojek team was able to visualize and identify supply and demand issues. They discovered that the areas with the highest discrepancy between supply and demand came from train stations. Often there were far more booking requests than there were available drivers. Since using Google Cloud's big data and machine learning products, the Gojek team has been able to actively monitor requests to ensure the drivers are in the areas with the highest demand. This brings faster bookings for riders and more work for the drivers.

QWIKLABS Section:

Hello and welcome. I'm Philipp Maier, Course Developer with Google Cloud. And this is a brief tutorial on using quick labs in this course. I'm about to show you the interactive hands-on labs platform called QWIKLABS, which is part of Google Cloud. QWIKLABS allows you to get practical hands-on experience with Google Cloud. And proficiency with Google account credentials so that you can access the cloud console at no cost. The first step is to lock into coursera in an incognito window, depending on your browser, it might also be called private browsing or in private browsing. Locking into Coursera from a private window, ensures that you don't accidentally use your own Google account. Well accessing the cloud console. We don't want you to get any unexpected bills at the end of the month. Check out the links in the download section of this video for different browser support articles. Once logged into coursera using an incognito window, return to your course and go to the lab activity page.

Play video starting at :1:1 and follow transcript

1:01

If prompted, you want to accept the honour code, and enter your name.

Play video starting at :1:7 and follow transcript

1:07

Then click open tool to open that lab in a new tab. And within this new tab, click Start lab and wait for the connection details to be displayed on the left. So for each lab you will have a timer so you can see here with D remaining access time, your lab will automatically end when the timer runs out.

Play video starting at :1:32 and follow transcript

1:32
Now you want to click Open Google console And then sign in with the username and password provided in this connections Detail page. So I'm going to grab the username here and paste that in. And this is going to be different for each lab. And then also grab the password and paste that in there as well.

Play video starting at :1:58 and follow transcript

1:58
Now quick labs creates a new account for you. Each time you launch a lab, therefore, you need to click through the initial account setup windows. So we have a new account here. So I'm going to accept asking for a backup. I don't need that. I'm just going to confirm. And once I am in the Cloud Console, I'm also going to have to accept the terms and services. So let's just wait for that to load up.

Play video starting at :2:32 and follow transcript

2:32
And then once I'm in that project also are in the account, I can verify that I'm using the right project and I'll show you how to do that.

Play video starting at :2:40 and follow transcript

2:40
So I'm just waiting for the terms of services, agree and then continue.

Play video starting at :2:47 and follow transcript

2:47
So up here, you can see the current project ID that I'm using. And it's a good thing to double check that this project ID actually matches what you have in quick labs, and we can see that here. This is the same project ID. And sometimes you actually have multiple project IDs, and maybe even multiple users, if you're trying to do specific sharing between users or adding removing permissions and so on.

Play video starting at :3:14 and follow transcript

3:14
I can also see that my username in here is not my actual email. This is the account that quick labs could have created for me. So it's important to make sure that these match, again the credentials that are in here, so that you're not using your own account.

Play video starting at :3:33 and follow transcript

3:33
Now some labs track your work within the quick labs provided Google Cloud project. If this is enabled, you will see a score in the top right corner of the quick labs window. As you can see here. Now your score increases as objectives are met. And you can click on the score to view the individual steps to be scored as you can see here right now.

Play video starting at :3:56 and follow transcript

3:56
So I'm going to go complete these and then come back. Now that I have completed the lab, I can see my score has been updated and I'm ready to click End lab.

Play video starting at :4:10 and follow transcript

4:10
And then confirm with OK.

Play video starting at :4:13 and follow transcript

I can now give this some feedback. And once I click and lab D QWIKLABS provides a project and any resources within that project will be deleted.
Play video starting at :4:24 and follow transcript

4:24
So I can now close the quick labs lab page.
Play video starting at :4:28 and follow transcript

4:28
So here I'm on the lab page. And if I can scroll up, you'll see that I have successfully completed the lab. And if I go to the grades section of the course, you'll also see that I have a full score for this lap. That's it for this tutorial. Remember to launch coursera in an incognito window and use the quick lead provider credentials to sign in to the Cloud Console. Good luck with the labs and enjoy the rest of this course.

ML Learning and Google Cloud:

In previous sections of this course, you learned about many data engineering tools available from Google Cloud. Now let’s switch our focus to machine learning. In this section, we’ll explore the different options Google Cloud offers for building machine learning models. Additionally, we will explain how a product called Vertex AI can help solve machine learning challenges. So you might be wondering, “Why should I trust Google for artificial intelligence and machine learning?” Google is an AI-first company, and is recognized as a leader across industries because of its contributions in the fields of artificial intelligence and machine learning. In 2022 Google was recognized as a leader in the Gartner Magic Quadrant for Cloud AI Developer services, and in recent years has also received recognition in numerous annual industry awards and reports. And at Google we’ve been implementing artificial intelligence for over ten years into many of our critical products, systems, and services. For example, have you ever noticed how Gmail automatically suggests three responses to a received message? This feature is called Smart Reply, which uses artificial intelligence to predict how you might respond. Behind this intelligence is AI technology known as natural language processing, which is just one example of an impressive list of technologies that Google scientists and engineers are working on. We’ll explore these in more depth later in the course. The goal of these technologies is not for exclusive use to only benefit Google customers. The goal is to enable every company to be an AI company by reducing the challenges of AI model creation to only the steps that require human judgment or creativity. So for workers in the travel and hospitality field, this might mean using AI and ML to improve aircraft scheduling or provide customers with dynamic pricing options. For retail-sector employees, it might mean using AI and ML to leverage predictive inventory planning. The potential solutions are endless. What are the problems in your business that artificial intelligence and machine learning might help you solve? Take a moment to think about this question before continuing to the next video.

Good Machine Learning models require lots of high-quality training data. You should aim for hundreds of thousands of records to train a custom model. If you don't have that kind of data, pre-built APIs are a great place to start. Pre-built APIs are offered as services. In many cases they can act as building blocks to create the application you want without expense or complexity of creating your own models. They save the time and effort of building, curating, and training a new dataset so you can just jump right ahead to predictions. So, what are some of the pre-built APIs? Let’s explore a short list. The Speech-to-Text API converts audio to text for data processing. The Cloud Natural Language API recognizes parts of speech called entities and sentiment. The Cloud Translation API converts text from one language to another. The

Text-to-Speech API converts text into high quality voice audio. The Vision API works with and recognizes content in static images. And the Video Intelligence API recognizes motion and action in video. And Google has already done a lot of work to train these models using Google datasets. For example, the Vision API is based on Google's image datasets, the Speech-to-Text API is trained on YouTube captions, and the Translation API is built on Google's neural machine translation technology. You'll recall that how well a model is trained depends on how much data is available to train it. As you might expect, Google has a lot of images, text, and ML researchers to train it's pre-built models. This means less work for you. Let's take a minute and try out the Vision API in a browser. (Provide an image.) Start by navigating to cloud.google.com/vision in Chrome, and then Scroll down to try the API by uploading an image. You can actually experiment with each of the ML APIs in a browser. When you're ready to build a production model, you'll need to pass a JSON object request to the API and parse what it returns.

Automated Machine Learning:

To understand AutoML, which is short for automated machine learning, let's briefly look at how it was built. If you've worked with ML models before, you know that training and deploying ML models can be extremely time consuming, because you need to repeatedly add new data and features, try different models, and tune parameters to achieve the best result. To solve this problem, when AutoML was first announced in January of 2018, the goal was to automate machine learning pipelines to save data scientists from manual work, such as tuning hyperparameters and comparing against multiple models. But how could this be done? Well, machine learning is similar to human learning. It all starts with gathering the right information. For AutoML, two technologies are vital. The first is known as transfer learning. With transfer learning, you build a knowledge base in the field. You can think of this like gathering lots of books to create a library. Transfer learning is a powerful technique that lets people with smaller datasets, or less computational power, achieve state-of-the-art results by taking advantage of pre-trained models that have been trained on similar, larger data sets. Because the model learns via transfer learning, it doesn't have to learn from scratch, so it can generally reach higher accuracy with much less data and computation time than models that don't use transfer learning. The second technology is neural architecture search. The goal of neural architecture search is to find the optimal model for the relevant project. Think of this like finding the best book in the library to help you learn what you need to. AutoML is powered by the latest machine-learning research, so although a model performs training, the AutoML platform actually trains and evaluates multiple models and compares them to each other. This neural architecture search produces an ensemble of ML models and chooses the best one. Leveraging these technologies has produced a tool that can significantly benefit data scientists. One of the biggest benefits is that it's a no-code solution. That means it can train high-quality custom machine learning models with minimal effort and requires little machine learning expertise. This allows data scientists to focus their time on tasks like defining business problems or evaluating and improving model results. Others might find AutoML useful as a tool to quickly prototype models and explore new datasets before investing in development. This might mean using it to identify the best features in a dataset, for example. So, how does AutoML work exactly? AutoML supports four types of data: image, tabular, text, and video. For each data type, AutoML solves different types of problems, called objectives. To get started, upload your data into AutoML. It can come from Cloud Storage, BigQuery, or even your local machine. From there, inform AutoML of the problems you want to solve. Some problems may sound similar to those mentioned in pre-built APIs. However the major difference is that pre-built APIs use pre-built machine learning models, while AutoML uses custom-built models. In AutoML, you use your own data to train the machine learning model and then apply the trained model to predict your goal. For image data: You can use a classification

model to analyze image data and return a list of content categories that apply to the image. For example, you could train a model that classifies images as containing a dog or not containing a dog, or you could train a model to classify images of dogs by breed. You can also use an object detection model to analyze your image data and return annotations that consist of a label and bounding box location for each object found in an image. For example, you could train a model to find the location of the dogs in image data. For tabular data: You can use a regression model to analyze tabular data and return a numeric value. For example, you could train a model to estimate a house's value or rental price based on a set of factors such as location, size of the house, and number of bedrooms. You can use a classification model to analyze tabular data and return a list of categories. For example, you could train a model to classify different types of land into high, median, and low potentials for commercial real estate. And a forecasting model can use multiple rows of time-dependent tabular data from the past to predict a series of numeric values in the future. For example, you could use the historical plus the economic data to predict what the housing market will look like in the next five years. For text data: You can use a classification model to analyze text data and return a list of categories that apply to the text found in the data. For example, you can classify customer questions and comments to different categories and then redirect them to corresponding departments. An entity extraction model can be used to inspect text data for known entities referenced in the data and label those entities in the text. For example, you can label a social media post in terms of predefined entities such as time, location, and topic, etc. This can help with online search, similar to the concept of a hashtag, but created by machine. And a sentiment analysis model can be used to inspect text data and identify the prevailing emotional opinion within it, especially to determine a writer's comment as positive, negative, or neutral. And finally, for video data: You can use a classification model to analyze video data and return a list of categorized shots and segments. For example, you could train a model that analyzes video data to identify whether the video is of a soccer, baseball, basketball, or football game. You can use an object tracking model to analyze video data and return a list of shots and segments where these objects were detected. For example, you could train a model that analyzes video data from soccer games to identify and track the ball. And an action recognition model can be used to analyze video data and return a list of categorized actions with the moments the actions happened. For example, you could train a model that analyzes video data to identify the action moments involving a soccer goal, a golf swing, a touchdown, or a high five. In reality, you may not be restricted to just one data type and one objective but rather need to combine multiple data types and different objectives to solve a business problem. AutoML is a powerful tool that can help across these different data types and objectives.

Custom Training:

We've explored the options google cloud provides to build machine learning models using BigQuery ML, Pre-built APIs and AutoML. Now let's take a look at the last option custom training.
Play video starting at ::13 and follow transcript

0:13
If you want to code your machine learning model, you can use this option by building a custom training solution with Vertex AI Workbench. Workbench is a single development environment for the entire data science workflow from exploring, to training and then deploying a machine learning model with code. Before any coding begins, you need to determine what environment you want, your ML training code to use. There are two options a pre-built container or a custom container. Imagine that a container is a kitchen. A pre-built container would represent a fully furnished room with cabinets and appliances which represent the dependencies that includes all

the cookware, which represents the libraries, you need to make a meal. So if your ML training needs a platform like TensorFlow, Pytorch, Scikit-learn or XGboost and Python code to work with the platform, a pre built container is probably your best solution. A custom container alternatively is like an empty room with no cabinets, appliances or cookware. You define the exact tools, you need to complete the job.

Vertex AI:
For years, Google has invested time and resources into developing big data and AI. Google had developed key technologies and products from Scikit learn as a Google Summer Coding project back in 2007 to Vertex AI today. As an AI first company, Google has applied AI technologies to many of its products and services like Gmail, Google Maps, Google Photos and Google Translate just to name a few. But developing these technologies doesn't come without challenges, especially when it involves developing machine learning models and putting them into production. Some traditional challenges include determining how to handle large quantities of data. Determining the right machine learning model to train the data and harnessing the required amount of computing power. Then there are challenges around getting ML models into production. Production requires scalability monitoring and continuous integration and continuous delivery or deployment. These challenges can make projects fail. In fact, according to Gartner, only half of enterprise ML projects get past the pilot phase. And finally, there are ease of use challenges. Many tools in the market require advanced coding skills, which can take a data scientists focus away from model configuration. And without a unified workflow, data scientists often have difficulties finding tools. Google's solution to many of the production and ease of use challenges is Vertex AI. A unified platform that brings all the components of the machine learning ecosystem and work flow together. So what exactly does a unified platform mean? In the case of Vertex AI, it means having one digital experience to create, deploy and manage models over time and at scale. For example, during the data readiness stage users can upload data from wherever it's stored. Cloud storage, Big Query or a local machine. Then during the feature already in this stage, users can create features which are the processed data that will be put into the model and then share them with others using the feature store. After that, it's time for training and hyperparameter tuning. This means that when the data is ready, users can experiment with different models and adjust hyperparameters. And finally, during deployment and model monitoring, users can set up the pipeline to transform the model into production by automatically monitoring and performing continuous improvements. And to refer back to the different options we explored earlier, Vertex AI allows users to build machine learning models with either AutoML, a code list solution or custom training, a code based solution. AutoML is easy to use and lets data scientists spend more time turning business problems into ML solutions. While custom training enables data scientists to have full control over the development environment and process. Being able to perform such a wide range of tasks in one unified platform has many benefits. This can be summarized with four Ss. It's seamless. Vertex AI provides a smooth user experience from uploading, preparing data all the way to model training and production. It's scalable. The machine learning operations, ML ops provided by Vertex AI helps to monitor and manage the ML production and therefore scale the storage and computing power automatically. It's sustainable. All of the artifacts and features created using Vertex AI can be reused and shared. And its speedy. Vertex AI produces models that have 80% fewer lines of code than competitors.

AI Solutions:
Now that you've explored the four different options available to create machine learning models with Google Cloud, let's take a few minutes to explore Google Cloud's artificial intelligence solution portfolio. It can be visualized with three layers. The bottom layer is the AI foundation,

and includes the Google Cloud infrastructure and data. The middle layer represents the AI development platform, which includes the four ML options you just learned about: AutoML and custom training, which are offered through Vertex AI, and pre-built APIs and BigQuery ML. The top layer represents AI solutions, for which there are two groups, horizontal solutions and industry solutions. Horizontal solutions usually apply to any industry that would like to solve the same problem. Examples include Document AI and CCAI. Document AI uses computer vision and optical character recognition, along with natural language processing, to create pretrained models to extract information from documents. The goal is to increase the speed and accuracy of document processing to help organizations make better decisions faster, while reducing costs. Another example of a horizontal solution is Contact Center AI, or CCAI. The goal of CCAI is to improve customer service in contact centers through the use of artificial intelligence. It can help automate simple interactions, assist human agents, unlock caller insights, and provide information to answer customer questions. And the second group is vertical, or industry solutions. These represent solutions that are relevant to specific industries. Examples include: Retail Product Discovery, which gives retailers the ability to provide Google-quality search and recommendations on their own digital properties, helping to increase conversions and reduce search abandonment, Google Cloud Healthcare Data Engine, which generates healthcare insights and analytics with one end-to-end solution, and Lending DocAI, which aims to transform the home loan experience for borrowers and lenders by automating mortgage document processing. You can learn more about Google Cloud's growing list of AI solutions at cloud.google.com/solutions/ai.

in the previous section of this course you explore the machine learning options available on Google Cloud now let's switch our Focus to the machine learning workflow with vertex AI from data preparation to model training and finally model deployment vertex AI Google's AI platform provides developers and data scientists one unified environment to build custom ml models this process is actually not too different from serving food in a restaurant starting with preparing raw ingredients through to serving dishes to a table later in the section you'll get Hands-On practice building a machine learning model end to end using automl on vertex AI but before we get into the details let's look at the basic differences between machine learning and traditional programming in traditional programming simply put one plus one equals two data Plus rules otherwise known as algorithms lead to answers and with traditional programming a computer can only follow the algorithms that a human has set up but what if we're just too lazy to figure out all the algorithms or what if the algorithms are too complex to figure out this is where machine learning comes in with machine learning you feed a machine a large amount of data along with answers that you would expect a model to conclude from the data then you select a machine learning model from there you expect the machine to learn from the provided data and examples to solve the puzzle on its own so instead of telling a machine how to do addition you give it pairs of numbers and the answers for example 1 1 and 2 2 3 and 5. you then ask it to figure out how to do addition on its own but how is it possible that a machine can actually learn to solve puzzles for machine learning to be successful you'll need lots of storage like what's available with cloud storage and the ability to make fast calculations like with cloud computing there are many practical examples of this capability for example by feeding Google photos lots of pictures with tags you can teach the software to associate and then automatically attach tags to new pictures tags can then be used for search functionality or even to automatically create photo albums can you come up with any other examples to apply machine learning capabilities take a moment to think about it

there are three key stages to this learning process the first is data preparation which includes two steps data uploading and feature engineering you'll be introduced to feature Engineering in the next lesson a model needs a large amount of data to learn from data used in machine learning can either be real-time streaming or batch data and it can either be structured which is numbers and text normally saved in tables or unstructured which is data that can't be put into tables like images and videos the second stage is model training a model needs a tremendous amount of iterative training this is when training and evaluation form a cycle to train data then evaluate the data and then train the data some more the third and final stage is model serving a model needs to actually be used in order to predict results this is when the machine learning model is deployed monitored and managed if you don't move an ml model into production then it has no use and remains only a theoretical model now it was mentioned earlier that the machine learning workflow on vertex AI is not too different from serving food in a restaurant so if you compare these steps to Running a Restaurant data preparation is when you prepare the raw ingredients model training is when you experiment with different recipes and metal serving is when you finalize the menu to serve the meal to lots of hungry customers it's important to note that an ml workflow isn't linear it's iterative for example during model training you may need to return to dig into the raw data and generate more useful features to feed the model when monitoring the model during model serving you might find data drifting or the accuracy of your prediction might subtly drop you might need to check the data sources and adjust the model parameters fortunately these steps can be automated with machine learning operations or ml Ops we'll go into more detail on this soon so how does vertex AI support this workflow you'll recall that vertex AI provides two options to build machine learning models automl which is a codeless solution and custom Training which is a code-based solution vertex AI provides many features to support the ml workflow all of which are accessible through either automl or vertex AI workbench examples include feature store which provides a centralized repository for organizing storing and serving features to feed to training models vizier which helps you tune hyper parameters in complex machine learning models explainable AI which helps interpret training performance and model behaviors and pipelines which help you automate and monitor the ml production line

Data Preparation:
Now let’s look closer at an AutoML workflow. The first stage of the AutoML workflow is data preparation. During this stage, you must upload data and then prepare the data for model training with feature engineering. When you upload a dataset in the Vertex AI user interface, you’ll need to provide a meaningful name for the data and then select the data type and objective. AutoML allows four types of data: image, tabular, text, and video. To select the correct data type and objective, you should: Start by checking data requirements. We’ve included a link to these requirements in the resources section of this course. Next, you’ll need to add labels to the data if you haven’t already. A label is a training target. So, if you want a model to distinguish a cat from a dog, you must first provide sample images that are tagged—or labeled—either cat or dog. A label can be manually added, or it can be added by using Google’s paid label service via the Vertex console. These human labellers will manually generate accurate labels for you. The final step is to upload the data. Data can be uploaded from a local source, BigQuery, or Cloud Storage. You will practice these steps in the lab. After your data is uploaded to AutoML, the next step is preparing the data for model training with feature engineering. Imagine you’re in the kitchen preparing a meal. Your data is like your ingredients, such as carrots, onions, and tomatoes. Before you start cooking, you'll need to peel the carrots, chop the onions, and rinse the tomatoes. This is what feature engineering is like: the data must be processed before the model starts training. A feature, as we discussed in the BigQuery module, refers to a factor that contributes to the prediction. It’s an independent variable in statistics or a column in a table. Preparing features can be both challenging and tedious. To help, Vertex AI has a function called

Feature Store. Feature Store is a centralized repository to organize, store, and serve machine learning features. It aggregates all the different features from different sources and updates them to make them available from a central repository. Then, when engineers need to model something, they can use the features available in the Feature Store dictionary to build a dataset. Vertex AI automates the feature aggregation to scale the process. So what are the benefits of Vertex AI Feature Store? First, features are shareable for training or serving tasks. Features are managed and served from a central repository, which helps maintain consistency across your organization. Second, features are reusable. This helps save time and reduces duplicative efforts, especially for high-value features. Third, features are scalable. Features automatically scale to provide low-latency serving, so you can focus on developing the logic to create the features without worrying about deployment. And fourth, features are easy to use. Feature Store is built on an easy-to-navigate user interface.

Model Training:
Now that our data is ready, which, if we return to the cooking analogy is our ingredients, it's time to train the model. This is like experimenting with some recipes. This stage involves two steps: model training, which would be like cooking the recipe, and model evaluation, which is when we taste how good the meal is. This process might be iterative. Before we get into more details about this stage, let's pause to clarify two terms: artificial intelligence and machine learning. Artificial intelligence, or AI, is an umbrella term that includes anything related to computers mimicking human intelligence. For example, in an online word processor, robots performing human actions all the way down to spell check. Machine learning is a subset of AI that mainly refers to supervised and unsupervised learning. You might also hear the term deep learning, or deep neural networks. It's a subset of machine learning that adds layers in between input data and output results to make a machine learn at more depth. So, what's the difference between supervised and unsupervised learning? Supervised learning is task-driven and identifies a goal. Unsupervised learning, however, is data-driven and identifies a pattern. An easy way to distinguish between the two is that supervised learning provides each data point with a label, or an answer, while unsupervised does not. For example, if we were given sales data from an online retailer, we could use supervised learning to predict the sales trend for the next couple of months and use unsupervised learning to group customers together based on common characteristics. There are two major types of supervised learning: The first is classification, which predicts a categorical variable, like using an image to tell the difference between a cat and a dog. The second type is a regression model, which predicts a continuous number, like using past sales of an item to predict a future trend. There are three major types of unsupervised learning: The first is clustering, which groups together data points with similar characteristics and assigns them to "clusters," like using customer demographics to determine customer segmentation. The second is association, which identifies underlying relationships, like a correlation between two products to place them closer together in a grocery store for a promotion. And the third is dimensionality reduction, which reduces the number of dimensions, or features, in a dataset to improve the efficiency of a model. For example, combining customer characteristics like age, driving violation history, or car type, to create an insurance quote. If too many dimensions are included, it can consume too many compute resources, which might make the model inefficient. Although Google Cloud provides four machine learning options, with AutoML and pre-built APIs you don't need to specify a machine learning model. Instead, you'll define your objective, such as text translation or image detection. Then on the backend, Google will select the best model to meet your business goal. With the other two options, BigQuery ML and custom training, you'll need to specify which model you want to train your data on and assign something called hyperparameters. You can think of hyperparameters as user-defined knobs in a machine that helps guide the machine learning process. For example, one parameter is a learning rate, which

is how fast you want the machine to learn. With AutoML, you don't need to worry about adjusting these hyperparameter knobs because the tuning happens automatically in the back end. This is largely done by a neural architect search, which finds the best fit model by comparing the performance against thousands of other models.

Model Evaluation:
While we are experimenting with a recipe, we need to keep tasting it constantly to make sure it meets our expectations. This is the model evaluation portion of the model training stage. Vertex AI provides extensive evaluation metrics to help determine a model's performance. Among the metrics are two sets of measurements. The first is based on the confusion matrix, for example recall and precision. The second is based on feature importance, which we'll explore later in this section of the course. A confusion matrix is a specific performance measurement for machine learning classification problems. It's a table with combinations of predicted and actual values. To keep things simple we assume the output includes only two classes. Let's explore an example of a confusion matrix. The first is a true positive combination, which can be interpreted as, "The model predicted positive, and that's true." The model predicted that this is an image of a cat, and it actually is. The opposite of that is a true negative combination, which can be interpreted as, "The model predicted negative, and that's true." The model predicted that a dog is not a cat, and it actually isn't. Then there is a false positive combination, otherwise known as a Type 1 Error, which can be interpreted as, "The model predicted positive, and that's false." The model predicted that a dog is a cat but it actually isn't. Finally, there is the false negative combination, otherwise known as a Type 2 Error, which can be interpreted as, "The model predicted negative, and that's false." The model predicted that a cat is not a cat, but it actually is. A confusion matrix is the foundation for many other metrics used to evaluate the performance of a machine learning model. Let's take a look at the two popular metrics, recall and precision, that you will encounter in the lab. Recall refers to all the positive cases, and looks at how many were predicted correctly. This means that recall is equal to the true positives, divided by the sum of the true positive and false negatives. Precision refers to all the cases predicted as positive, and how many are actually positive. This means that precision is equal to the true positives, divided by the sum of the true positive and false positives. Imagine you're fishing with a net. Using a wide net, you caught both fish and rocks: 80 fish out of 100 total fish in the lake, plus 80 rocks. The recall in this case is 80%, which is calculated by the number of fish caught, 80, divided by the total number of fish in the lake, 100. The precision is 50%, which is calculated by taking the number of fish caught, 80, and dividing it by the number of fish and rocks collected, 160. Let's say you wanted to improve the precision, so you switched to a smaller net. This time you caught 20 fish and 0 rocks. The recall becomes 20% (20 out of 100 fish collected) and the precision becomes 100% (20 out of 20 total fish and rocks collected). Precision and recall are often a trade-off. Depending on your use case, you may need to optimize for one or the other. Consider a classification model where Gmail separates emails into two categories: spam and not-spam. If the goal is to catch as many potential spam emails as possible, Gmail may want to prioritize recall. In contrast, if the goal is to only catch the messages that are definitely spam without blocking other emails, Gmail may want to prioritize precision. In Vertex AI, the platform visualizes the precision and the recall curve so they can be adjusted based on the problem that needs solving. You'll get the opportunity to practice adjusting precision and recall in the AutoML lab. In addition to the confusion matrix and the metrics generated to measure model effectiveness such as recall and precision, the other useful measurement is feature importance. In Vertex AI, feature importance is displayed through a bar chart to illustrate how each feature contributes to a prediction. The longer the bar, or the larger the numerical value associated with a feature, the more important it is. This information helps decide which features are included in a machine learning model to predict the goal. You will observe the feature importance chart in the lab as well. Feature importance is just one

example of Vertex AI’s comprehensive machine learning functionality called Explainable AI. Explainable AI is a set of tools and frameworks to help understand and interpret predictions made by machine learning models.

Model Deployment and Monitoring:
The recipes are ready and now it's time to serve the meal. This represents the final stage of the machine learning workflow. Model serving. Model serving consists of two steps. First, model deployment, which we can compare to serving the meal to a hungry customer and second, model monitoring, which we can compare to checking with the waitstaff to ensure that the restaurant is operating efficiently. It's important to note that model management exists throughout this whole workflow to manage the underlying machine-learning infrastructure. This lets data scientists focus on what to do rather than how to do it. Machine learning operations or MLOps play a big role. MLOps combines machine-learning development with operations and apply similar principles from DevOps to machine-learning models, which is short for development and operations. MLOps aims to solve production challenges related to machine learning. In this case, this refers to building an integrated machine learning system, and operating it in production. These are considered to be some of the biggest pinpoints by the ML practitioners community because both data and code are constantly evolving in machine learning. Practicing MLOps means advocating for automation and monitoring at each step of the ML system construction. This means adopting a process to enable continuous integration, continuous training, and continuous delivery. What does MLOps have to do with model serving? Well, let's start with model deployment, which is the exciting time when a model is implemented. In a restaurant analogy, it's when the food is put on the table for the customer to eat. MLOps provides a set of best practices on the backend to automate this process. There are three options to deploy a machine-learning model. The first is to deploy to an endpoint. This option is best when immediate results with low latency are needed, such as making instant recommendations based on a user's browsing habits whenever they're online. A model must be deployed to an endpoint before that model can be used to serve real-time predictions. The second option is to deploy using batch prediction. This option is best when no immediate response is required and accumulated data should be processed with a single request. For example, sending out new ads every other week based on the user's recent purchasing behavior and what's currently popular on the market. The final option is to deploy using offline prediction. This option is best when the model should be deployed in a specific environment off the cloud. In the lab, you'll practice predicting with an endpoint. Now let's shift our focus to model monitoring. The backbone of MLOps and Vertex AI is a tool called Vertex AI Pipelines. It automates, monitors, and governs machine-learning systems by orchestrating the workflow in a serverless manner. Imagine you're in a production control room and Vertex AI Pipelines is displaying the production data onscreen. If something goes wrong, it automatically triggers warnings based on a predefined threshold. With Vertex AI Workbench, which is a Notebook tool, you can define your own pipeline. You can do this with prebuilt pipeline components, which means that you primarily need to specify how the pipeline is put together using component as building blocks. It's with these final two steps, model deployment and model monitoring, that we complete our exploration of the machine learning workflow. The restaurant is open and operating smoothly. Bon appetit.

Predicting precision recall:
Well done on completing the AutoML lab! You’ve now had a chance to use Vertex AI to build a machine learning model without writing a single line of code. Let’s take a few moments to review the results of the lab, starting with the confusion matrix. But before that begins, please pause to consider the matrix results for yourself. The true positives were 100%. This represents the percentage of people the model predicted would repay their loan who actually did pay it back.

The true negatives were 87%. This represents the percentage of people the model predicted would not repay their loan who indeed did not pay it back. The false negatives were 0%. This represents the percentage of people the model predicted would not repay their loan, but who actually did pay it back. And finally, the false positives were 13%. This represents the percentage of people the model predicted would repay their loan, but who actually did not pay it back. As a general principle, it's good to have high true positives and true negatives, and low false positives and false negatives. However, how high or low they need to be really depends on the business goals you're looking to achieve. There are different ways to improve the performance of a model, which might include using a more accurate data source, using a larger dataset, choosing a different type of ML model, or tuning the hyperparameters. Let's also review the precision-recall curve from the AutoML lab. The confidence threshold determines how a machine learning model counts the positive cases. A higher threshold increases the precision, but decreases recall. A lower threshold decreases the precision, but increases recall. Moving the threshold to zero produces the highest recall of 100%, and the lowest precision of 50%. So, what does that mean? That means the model predicts that 100% of loan applicants will be able to repay a loan they take out. However, in actuality, only 50% of people were able to repay the loan. Using this threshold to identify the default cases in this example can be risky, because it means that you're only likely to get half of the loan investment back. Now let's move to the other extreme by moving the threshold to 1. This will produce the highest precision of 100% with the lowest recall of 1%. What does this mean? It means that of all the people who were predicted to repay the loan, 100% of them actually did. However, you rejected 99% of loan applicants by only offering loans to 1% of them. That's a pretty big loss of business for your company. These are both extreme examples, but it's important that you always try to set an appropriate threshold for your model.

Summary:
Congratulations on completing the Big Data and Machine Learning Fundamentals course! We hope that you learned some valuable information from this course that will help advance your career. Throughout the course, we introduced a number of products and technologies to support Google's data-to-AI lifecycle. Let's do a final review of the main concepts presented. In the first section of the course, you were introduced to the Google Cloud infrastructure and Google's big data and machine learning products. Of the three layers of the Google Cloud infrastructure, you explored the middle and top layers. On the middle layer sit compute and storage. Google Cloud decouples compute and storage so they can scale independently based on need. And on the top layer sit the big data and machine learning products, which enable you to perform tasks to ingest, store, process, and deliver business insights, data pipelines, and machine learning models. In the second section of the course, you explored data engineering for streaming data. This included how to build a streaming data pipeline– from ingestion with Pub/Sub, to processing with Dataflow, to visualization using Looker and Looker Studio. After that, in the third section of the course, you were introduced to BigQuery, which is Google's fully-managed data warehouse. BigQuery provides two services in one: storage plus analytics. You also learned about BigQuery ML, the machine learning tool used for developing machine learning models directly in BigQuery. In the fourth section of the course, you explored the options available to build and deploy machine learning models with Google Cloud. If you're familiar with SQL and already have data in BigQuery, you can use BigQuery ML to develop machine learning models. If you have little ML experience, using pre-built APIs is likely the best choice. Pre-built APIs address common perceptual tasks such as vision, video, and natural language. They're ready to use without any ML expertise or model development effort. If you want to build custom models with your own training data while spending minimal time coding, then AutoML is a great choice. AutoML provides a code-less solution to enable you to focus on business problems instead of the underlying model architecture and ML provisioning. And if you want full control of the machine

learning workflow, Vertex AI custom training lets you train and serve custom models with code on Vertex Workbench. Using pre-built containers, you can leverage popular ML libraries, such as Tensorflow and PyTorch. Alternatively, you can build a custom container from scratch. In the final section of the course, you learned about the machine learning workflow using Vertex AI, a unified platform that brings all the components of the machine learning ecosystem and workflow together. The machine learning workflow comprises three stages. In stage one, data preparation, data is uploaded and feature engineering is applied. In stage two, model training, the model is trained and evaluated. And in stage three, model serving, the model is deployed and monitored. We hope that this course is just the beginning of your big data and machine learning journey. For more training and hands-on practice with data engineering and analytics, please refer to cloud.google.com/training/data-engineering-and-analytics. And for more training with machine learning and AI, please explore the options available at cloud.google.com/training/machinelearning-ai. And if you're interested in validating your expertise and showcasing your ability to transform businesses with Google Cloud technology, you might consider working toward a a Google Cloud certification. You can learn more about Google Cloud's certification offerings at cloud.google.com/certifications. Thanks for completing this course. We'll see you next time!