Google Cloud Big Data and Machine Learning Fundamentals

mardi 13 juin 2023 15:28

Big data and Machine learning on Google Cloud

Google Cloud infrastructure: 3 layers: 1) Networking & security,2)compute, storage, 3) Big data ML products

Compute:
- compute power:
  - Compute Engine: is an IaaS offering, or infrastructure as a service, which provides compute, storage,and Network resources virtually that are similar to physical data
  - Google Kubernetes Engine, or GKE:runs containerized applications in a cloud environment, as opposed to on an individual virtual machine, like Compute Engine. (con dependencies.)
  - App Engine, a fully managed PaaS offering, or platform as a service: PaaS offerings bind code to libraries. serverless execution environment. Cloud Functions is often
  - Cloud Run: a fully managed compute platform that enables you to run request or event-driven stateless workloads without having to worry about servers. It automa

- Google Photos: automatic video stabilization: takes an unstable video,stabilizes it to minimize movement.
- TPU introduced by Google 2016 to overcome CPU and GPU. TPU is application-specific integrated circuits (ASICs). domain-specific hardware

Storage
- Cloud (compute-storage decoupled) vs desktop (compute-storage coupled) computing
- fully managed database and storage services: Cloud Storage,Cloud Bigtable, Cloud SQL, Cloud Spanner, Firestore And BigQuery
- Goal=reduce the time and effort needed to store data (creating an elastic storage bucket directly in a web interface or through a command line for example on Cloud Storag
- Storage depends on data type, and business need.
- (un)Structured data = (non)tabular
  - Unstructured suited to Cloud Storage (also BigQuery),
- An object is an immutable piece of data consisting of a file of any format. You store objects in containers called buckets.All buckets are associated with a project, and you ca
- Each project, bucket, and object in Google Cloud is a resource in Google Cloud,
- Cloud Storage clases (suited for unstructured data):
  - Standard (best for frequently accessed, or hot data.stored for only brief periods of time)
  - Nearline (infrequently accessed data:data backups, long-tail multimedia content, or data archiving.)
  - Coldline (low-cost option for storing infrequently accessed data.once every 90 days)
  - Archive (lowest-cost option, used ideally for data archiving, online backup, and disaster recovery.access less than once a year,)
- Structured data:
  - transactional workloads (stem from Online Transaction Processing systems, which are used when fast data inserts and updates are required to build row-based record that impact only a few records.)
  - and analytical workloads (stem from Online Analytical Processing systems,which are used when entire datasets need to be read.require complex queries, for example,
- Then determine wether data will be accessed using SQL or not:
  - Transactional-SQL-local/regional scalability-Cloud SQL
  - Transactional-SQL-global scalability-Cloud Spanner
  - Transactional-NoSQL-FireStore
  - Analytical-SQL-BigQuery (analyze petabyte-scale datasets)
  - Analytical-NoSQL-Cloud BigTable (best for real-time, high-throughput applications that require only millisecond)

Big data:
- 2002: Google File System, GFS, was designed to handle data sharing and petabyte storage at scale.
- 2004: MapReduce
- 2008: Dremel: breaking the data into smaller chunks called shards, and then compressing them. then uses a query optimizer to share tasks between the many shards of dat BigQuery.
- 2010: Colossus, in 2010, which is a cluster-level file system and successor to the Google File System. It is a Platform as a Service (PaaS) that supports querying using ANSI SQ
- 2012: Spanner, in 2012, which is a globally available and scalable relational database.
- 2015: Pub/Sub, in 2015, which is a service used for streaming analytics and data integration pipelines to ingest and distribute data.
- 2015: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,and AutoML, as a suite of machine learning products.
- 2021: Vertex AI, a unified ML platform
- Unified and stable platform: Cloud Storage Dataproc Cloud Bigtable BigQuery Dataflow Firestore Pub/Sub Looker Cloud Spanner AutoML, and Vertex AI,
- Product categories along the data-to-AI workflow:
  - ingestion & process:to digest both real-time and batch data (Pub/Sub, Dataflow (streaming data processing), Dataproc, Cloud Data Fusion)
  - storage: (Cloud Storage, Cloud SQL(relational), Cloud Spanner(relational), Cloud Bigtable(NoSQL), and Firestore(NoSQL))
  - analytics: (BigQuery:a fully managed data warehouse that can be used to analyze data through SQL commands; Looker, and Looker Studio)
  - ML: ML development platform and the AI solutions. (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: Document AI, C Healthcare Data Engine

Example:
- A unicorn is a privately held startup business valued at over US$1 billion.
- For data latency: Dataflow (streaming data processing) + BigQuery (real-time business insights)
- For geospatial: Dataflow (a streaming event data pipeline).
  - driver locations ping Pub/Sub every 30 seconds, and
  - Dataflow would process the data (was able to automatically manage the number of workers processing the pipeline to meet demand)
  - The pipeline would aggregate the supply pings from the drivers against the booking requests.
  - This would connect to Gojek's notification system to alert drivers where they should go
- Conclusion:actively monitor requests to ensure that drivers are in the areas with the highest demand. This brings faster bookings for riders and more work for the drivers.

Data Engineering for streaming data

real-time data solution:
- Ingest streaming data using Pub/Sub (data ingestion is where large amounts of streaming data are received)
- Process the data with Dataflow, and
- Visualize the results with Looker and Looker Studio
- (In between data processing with Dataflow and visualization with Looker or Looker Studio, the data is normally saved and analyzed in a data warehouse such as BigQuery)

- design streaming pipelines with Apache Beam
- streaming vs batch processing (media streaming vs downloading):
  - Batch processing is when the processing and analysis happens on a set of stored data: Payroll and billing systems
  - Streaming data is a flow of data records generated by various data sources: fraud detection or intrusion detection. data is analyzed in near real-time
- 4Vs: data engineers and data scientists four major challenges = variety, volume, velocity, and veracity:
  - data could come in from a variety of different sources and in various formats
  - Veracity: data quality (Due to several data types and sources, big data often has many data dimensions)

- IoT devices challenge:
  - data can be streamed from many different methods and devices
  - it can be hard to distribute event messages (notification) to the right subscribers
  - data can arrive quickly and at high volumes.
  - ensuring services are reliable, secure, and perform as expected
- Pub/Sub:
  - a tool to handle distributed message-oriented architectures at scale
  - (Publisher/Subscriber, or publish messages to subscribers).
  - distributed messaging service
  - Pub/Sub's APIs are open, the service is global by default, and it offers end-to-end encryption
  - end-to-end encryption:
    - Pub/Sub reads, stores, broadcasts to any subscribers of this data topic that new messages are available
  - A central element of Pub/Sub is the topic.
- Designing streaming pipelines with Apache Beam:
  - streaming input sources + pipe data into a data warehouse (dataflow)
  - "Process" : extract, transform, and load data, or ETL.
  - pipeline design phase questions:
    - Will the pipeline code be compatible with both batch and streaming data? Or need to be refactored?
    - Will the pipeline code software development kit, or SDK, being used have all the transformations, mid-flight aggregations and windowing and be able to handle late data?
    - Are there existing templates or solutions that should be referenced?
  - pipeline design: Apache Beam: an open source, unified programming model to define and **execute data processing pipelines**, including ETL, batch, and stream processing
    - unified, which means it uses a single programming model for both batch and streaming data
    - It's portable, which means it can work on multiple execution environments, like Dataflow and Apache Spark
    - extensible, which means it allows you to write and share your own connectors and transformation libraries.
    - Apache Beam provides pipeline templates (Java, Python, or Go)
- Implementing streaming pipelines on Cloud Dataflow:
  - Questions:
    - How much maintenance overhead is involved?
    - Is the infrastructure reliable?
    - How is the pipeline scaling handled?
    - How can the pipeline be monitored?
    - Is the pipeline locked in to a specific service provider?
  - Dataflow is a fully managed service for executing Apache Beam pipelines within the Google Cloud
  - Dataflow is serverless and NoOps
    - NoOps: doesn't require management from an operations team, because maintenance, monitoring, and scaling are automated.
    - Serverless computing is a cloud computing execution model
  - Dataflow tasks :
    - optimizing a pipeline model's execution graph
    - schedules out distributed work
    - scaler
    - auto-heals any worker faults
    - rebalances efforts to most efficiently use its workers.
    - execution engine to process and implement data processing pipelines
  - you don't need to monitor all of the compute and storage resources that Dataflow manages,
  - Dataflow templates:
    - streaming templates (for processing continuous, or real-time, data): Pub/Sub to BigQuery, Pub/Sub to Cloud Storage, Datastream to BigQuery, Pub/Sub to M
    - batch templates (for processing bulk data, or batch load data.): BigQuery to Cloud Storage, Bigtable to Cloud Storage, Cloud Storage to BigQuery, Cloud Spar
    - utility templates (activities related to bulk compression, deletion, and conversion)

- Visualization with Looker:
  - Looker: semantic modeling layer on top of databases using Looker Modeling Language, or LookML
  - LookML defines logic and permissions independent from a specific database or a SQL language,
  - The Looker platform is 100% web-based, which makes it easy to integrate into existing workflows and share with multiple teams at an organization
  - Looker API, which can be used to embed Looker reports in other applications.
  - Looker features:
    - Dashboards: schedule its delivery through storage services like: Google Drive, Slack, Dropbox.
- Visualization with Data Studio:
  - Looker Studio: integrated into BigQuery: doesn\u2019t require support from an administrator to establish a data connection, which is a requirement with Looker
  - integrated into Google Analytics, Google Cloud billing dashboard
  - to create a Looker Studio dashboard:
    - choose a template (a pre-built template or a blank report)
    - link the dashboard to a data source.
- Lab introduction: Creating a streaming data pipeline for a Real-Time dashboard with Dataflow:
  - **gcloud** is the command-line tool for Google Cloud.
  - pre-installed on Cloud Shell and supports tab-completion
  - list the active account name: gcloud auth list
  - list the project ID: gcloud config list project
  - to create dataset: Google Cloud Shell or the Google Cloud Console.
  - Pub/Sub is an asynchronous global messaging service.
    - decoupling senders and receivers, it allows for secure and highly available communication between independently written applications
    - delivers low-latency, durable messaging
    - publisher applications and subscriber applications connect with one another through the use of a shared string called a **topic**

- Google maintains a few public Pub/Sub streaming data topics
- BigQuery is a serverless data warehouse
  - Tables in BigQuery are organized into datasets.
  - to create a new BigQuery dataset:
    - command-line tool (**Cloud Shell**):
      - Create dataset: bq --location=us-west1 mk taxirides
      - Create Table: bq --location=us-west1 mk \
      - --time_partitioning_field timestamp \
      - --schema ride_id:string,point_idx:integer,latitude:float,longitude:float,\
      - timestamp:timestamp,meter_reading:float,meter_increment:float,ride_status:string,\
      - passenger_count:integer -t taxirides.realtime
    - BigQuery Console UI:
  - Create a Cloud Storage bucket (to provide working space for your Dataflow pipeline)
    - Cloud Storage allows world-wide storage and retrieval of any amount of data at any time
    - Applications: serving website content, storing data for archival and disaster recovery, or distributing large data objects to users via direct download.
  - Set up a Dataflow Pipeline
    - set up a streaming data pipeline to read sensor data from Pub/Sub, compute the maximum temperature within a time window, and write this out to B
    - Restart the connection to the Dataflow API.
    - Create a new streaming pipeline
    - Analyze the taxi data using BigQuery:
      - SELECT * FROM taxirides.realtime LIMIT 10
    - Perform aggregations on the stream for reporting
    - Stop the Dataflow Job (to free up resources for your project)
    - Create a real-time dashboard
    - Create a time series dashboard
    - Summary:
      - used Pub/Sub to collect streaming data messages from taxis and feed it through your Dataflow pipeline into BigQuery
      - Pub/Sub, which can be used to ingest a large volume of IoT data from diverse resources in various formats.
      - Dataflow, a serverless, NoOps service, to process the data. 'Process' here refers to ETL (extract, transform, and load).

Big data with BigQuery:

- BigQuery's two main services, storage and analytics
- BigQuery is a fully managed data warehouse
  - A data warehouse is a large store, containing terabytes and petabytes of data gathered from a wide range of sources within an organization, that's used to **g**
  - A **data lake** is just a pool of **raw, unorganized, and unclassified** data, which has **no specified purpose**. A data warehouse on the other hand, contains structu be used for **advanced querying**
  - Being **fully managed** means that BigQuery **takes care of the underlying infrastructure**, so you can focus on using SQL queries to answer business questions, scalability, and security.
  - BigQuery Storage: to store petabytes of data. **1 petabyte** is equivalent to **11,000 movies at 4k** quality.
  - BigQuery analytics: **machine learning (**using SQL**), geospatial analysis, and business intelligence**,
  - BigQuery is a fully managed serverless solution, meaning that you don't need to worry about provisioning any resources or managing servers in the backend to answer your organization's questions in the frontend.
  - By **encryption at rest**, we mean encryption used to protect data that is stored on a **disk**, including solid-state drives, or backup media.
  - data warehouse solution architecture:
    - input data can be either real-time or batch data
    - streaming data, which can be either structured or unstructured, high speed, and large volume, Pub/Sub is needed to digest the data.
    - If it's batch data, it can be directly uploaded to Cloud Storage.
    - both pipelines (real-time or batch data) lead to Dataflow to process the data (ETL)
    - (real-time+Pub/Sub)(batch data+Cloud Storage)+Dataflow (ETL)+BigQuery(analytics,AI, and ML)+(business intelligence(BI) tools)(AI/ML tools)
    - The job of the analytics engine of BigQuery at the end of a data pipeline is to ingest all the processed data after ETL, store and analyze it, and possibly c visualization and machine learning.
    - **business analyst, BI developers**: If you prefer to work in spreadsheets, you can query both small or large BigQuery datasets directly from **Google Shee** operations like pivot tables.
    - **data scientist or machine learning engineer:** you can directly call the data from BigQuery through **AutoML** or **Workbench (**parts of Vertex AI**)**
    - BigQuery is like a common staging area for data analytics workloads
- Storage and analytics
  - BigQuery :**fully managed storage facility to load and store datasets**, and also a **fast SQL-based analytical engine**
  - The two services are connected by **Google's high-speed internal network**.
  - BigQuery can ingest datasets from : **internal** data(data saved directly in BigQuery), **external** data, **multi-cloud** data (data stored in **multiple cloud services**, s datasets.
  - After the data is stored in BigQuery, it's fully managed by BigQuery and is **automatically replicated**, **backed up**, and **set up to autoscale**
  - BigQuery also offers the option to **query external data sources**–like data stored in other **Google Cloud storage** services like **Cloud Storage**, or in other **Googl Spanner or Cloud SQL**
    - That means a raw CSV file in Cloud Storage or a Google Sheet can be used to write a query **without being ingested by BigQuery first**
  - **inconsistency** might result from **saving and processing data separately**. To avoid that risk, consider using **Dataflow** to build a **streaming data pipeline into** patterns to load data into BigQuery:
    - **batch** load, where source data is loaded into a BigQuery table in a single batch operation. (**one-time** operation or it can be **automated to occur on a sc**
      - A batch load operation can create a **new** table or **append** data into an existing table.
    - **streaming**, where **smaller batches of data are streamed continuously** so that the data is available for **querying in near-real time**
    - **generated** data, where SQL statements are used to insert rows into an existing table or to write the results of a query to a table.
    - **BigQuery is optimized for running analytic queries over large datasets**.
    - It can perform queries on **terabytes of data in seconds and on petabytes in minutes.**
    - BigQuery analytics features:
      - **Ad hoc analysis** using Standard SQL, the **BigQuery SQL** dialect.
      - **Geospatial analytics** using geography data types and Standard SQL geography functions.
      - Building **machine learning models** using **BigQuery ML**.

Building rich, interactive **business intelligence dashboards** using **BigQuery BI Engine**.

Query types**:**

BigQuery runs **interactive** queries, which means that the **queries are executed as needed**

batch queries, where **each query is queued** on your behalf and the query starts **when idle resources are available**, usually within a few minutes.

BigQuery demo - San Francisco bike share

- If no **hyphen** in the project name, no need to **back ticks** around the project name
- hold down the command key or the **Windows** key it'll highlight all the data sets in your query, so you can click on it
- Click on Query Table. Click on field names from Schema to add them into Query editor (no need to write names)
- More>Format
- Run

SELECT, FROM, AS, GROUP BY, ORDER BY, WHERE, DESC, LIMIT

Explore In Data Studio

data definition language (DDL): I want a creation statement inside of the actual code itself

Create dataset

CREATE OR REPLACE **TABLE** <dataset name>.<table name> AS

CREATE OR REPLACE **VIEW** <dataset name>.<view name> AS : If the original public dataset updates, the derived table will update

Introduction to BigQuery ML:

Traditional:

export data from **datastore** into an **IDE** (integrated development environment) such as **Jupyter Notebook or Google Colab**

transform the data and perform all your **feature engineering** steps **before you can feed it into a training mode**l.

build the model in TensorFlow, or a similar library, and train it **locally** on your computer or on a **virtual machine**.

To **improve the model performance**, you also need to go back and forth to **get more data and create new features**.

BigQuery:

- **Create a model** with a SQL statement
- Write a **SQL prediction query** and invoke ml.Predict
- evaluating the model,

Hyperparameters are the settings applied to a model **before the training starts**, like the learning rate

Choosing which type of ML model depends on your **business goal and the datasets**.

Supervised models are **task-driven and identify a goal**.

Goal: classification: logistic regression

goal :predict a number (regression): linear regression

unsupervised models are **data-driven and identify a pattern**

goal: identify patterns or clusters: cluster analysis

We recommend that you start with these options (logistic/linear regression), and use the results to **benchmark** to compare against more comple networks)(XGBoost,AutoML Tables, wide and deep DNNs), which may take more time and computing resources to train and deploy

machine learning operations (**ML Ops**): BigQuery ML supports features to **deploy, monitor, and manage the ML production**

Importing TensorFlow models for **batch prediction**

Exporting models from BigQuery ML for **online prediction**

hyperparameter tuning using **Vertex AI Vizier**

ML development: upload data, engineer feature, train model, evaluate result

Using BigQuery ML to predict customer lifetime value (LTV):

LTV: to estimate how much revenue or profit you can expect from a customer given their history and customers with similar patterns.

a **record or row** in the dataset is called an **example, an observation, or an instance**.

A **label** is a correct answer, and you know it's correct because it comes from historical data.

Depending on what you want to predict, a label can be either a **numeric** variable, which requires a **linear regression** model, or a **categorical** variable, w model.

Those columns are called **features**, or at least potential features.

Understanding the quality of the data in each column and working with teams to **get more features or more history** is often the hardest part of any ML pro

feature engineering: **combine or transform feature** columns

BigQuery ML **automatically does one-hot encoding** for categorical values.

BigQuery ML **automatically splits the dataset into training data and evaluation data**

BigQuery ML project phases:

phase 1: you **extract, transform, and load** data into BigQuery, if it isn't there already.

If you're already using other Google products, like YouTube for example, look out for easy **connectors** to get that data into BigQuery before you build y

You can **enrich your existing data warehouse with other data sources** by using **SQL joins**.

phase 2: you **select** and **preprocess** features.

You can use **SQL** to create the training dataset for the model to learn from

phase 3: you **create the model inside BigQuery**.

This is done by using the "**CREATE MODEL**" command.

Give it a **name**, specify the **model type**, and pass in a SQL query with your training dataset.

phase 4: after your model is trained, you can execute an **ML.EVALUATE** query to evaluate the performance of the trained model on your evaluation dataset.

phase 5: use it to **make predictions**.

invoke the **ml.PREDICT** command on your newly trained model to return with predictions and the model's confidence in those predictions.

BigQuery ML key commands:

CREATE MODEL: create a model. Models have OPTIONS (model type), which you can specify.

CREATE OR REPLACE MODEL: to overwrite an existing model

ML.WEIGHTS: inspect what a model learned

That value indicates **how important the feature is for predicting the result**, or label.

ML.EVALUATE: To evaluate the model's performance

ML.PREDICT: to make batch predictions

BigQuery ML commands for supervised models:

Labels: you need a field in your training dataset titled **LABEL**, or you need to specify which field or fields are your labels using the **input_label_cols** in yo

Feature: features are the data columns that are part of your **SELECT** statement after your CREATE MODEL statement.

After a model is trained, you can use the **ML.FEATURE_INFO** command to get **statistics and metrics** about that column for additional analysis.

model object: created in BigQuery that **resides in your BigQuery dataset**. You train many different models, which will all be objects stored under your **tables and views**. Model objects can display information for **when it was last updated** or **how many training runs** it completed

Model types (regression/classification): Creating a new model is as easy as writing CREATE MODEL, choosing a type, and passing in a training dataset.

Training progress: While the model is running, and even after it's complete, you can view training progress with **ML.TRAINING_INFO**.
inspect weights (ML.WEIGHTS): to see what the model learned about the importance of each feature as it relates to the label you're predicting.
ML.EVALUATE: how well the model performed against its evaluation dataset.
ML.PREDICT: getting predictions (through referencing your model name and prediction dataset)

Predicting Visitor Purchases with a Classification Model with BigQuery ML:

BQML: a feature in BigQuery where data analysts can **create, train, evaluate, and predict** with machine learning models with **minimal coding**

Create a BigQuery dataset to store models

the web analytics dataset has nested and repeated fields like ARRAYS which **need to be broken apart into separate rows** in your dataset. This is accomplishe

Machine learning options on google cloud:

Smart Reply: Gmail automatically suggests three responses to a received message. it uses artificial intelligence (natural language processing) to predict how you m
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 crea**
Options to build ML models:

a. **BigQuery ML**: uses SQL queries to create and execute machine learning models in BigQuery
b. **pre-built APIs**: lets you **leverage machine learning models that have already been built and trained by Google**, so you don't have to build your own machin enough training data or sufficient machine learning expertise in-house.
c. **AutoML**: is a **no-code solution**, so you can **build your own machine learning models** on **Vertex AI** through a **point-and-click interface.**
d. **custom training:** through which you can **code your very own** machine learning **environment, the training, and the deployment**, which gives you flexibility a pipeline.

Differences between options:

Data type: BigQuery ML only supports tabular data while the other three support tabular, image, text, and video
Training data size: Pre-built APIs do not require any training data, while BigQuery ML and custom training require a large amount of data
Machine learning and coding expertise: Pre-Built APIs and AutoML are user friendly with low requirements, while Custom training has the highest requirem understand SQL.
Flexibility to tune the hyperparameters: At the moment, you can't tune the hyperparameters with Pre-built APIs or AutoML, however, you can experiment BigQueryML and custom training.
Time to train the model: Pre-built APIs require no time to train a model because they directly use pre-built models from Google. The time to train a model f on the specific project.

Normally, custom training takes the longest time because it builds the ML model from scratch, unlike AutoML and BigQuery ML.

Which option:

- **BigQuery ML:** data engineers, data scientists, and data analysts are familiar with SQL and already have your data in BigQuery: **BigQuery ML** lets you develop
- pre-built APIs: business users or developers with little ML experience: pre-built APIs (vision, video, and natural language)
- AutoML: If your developers and data scientists want to **build custom models** with your **own training data** while spending **minimal time coding**, then AutoM code-less solution to enable you to focus on business problems instead of the underlying model architecture and ML provisioning.
- custom training: If your ML engineers and data scientists want **full control of ML workflow**, **Vertex AI** custom training lets you train and serve custom mode