PROFESSIONAL MACHINE LEARNING ENGINEER · CHEAT SHEET · CERTIFYDECK

Google Cloud Certified Professional Machine Learning Engineer

Dominios, pesos oficiales y habilidades clave del temario de Professional Machine Learning Engineer, para repasar antes del examen.

Nivel: Advanced (Professional) Duración: 120 min Fuente: guía de estudio oficial de Microsoft Learn

1. Architecting low-code AI solutions

13–13%

Developing ML models using BigQuery ML or AutoML on Gemini Enterprise Agent Platform

- Elige el tipo de modelo según el problema: LOGISTIC_REG, BOOSTED_TREE_* o AUTOML_CLASSIFIER para clasificar; LINEAR_REG o los regresores para valores continuos; KMEANS para segmentar sin etiquetas; MATRIX_FACTORIZATION para recomendaciones; ARIMA_PLUS (univariante, muchas series con TIME_SERIES_ID_COL) o ARIMA_PLUS_XREG (con variables externas) para previsiones. ARIMA_PLUS es la referencia barata y rápida frente a la previsión neuronal de AutoML.
- BigQuery ML preprocesa solo (media para NULL, estandarización, one-hot y componentes de TIMESTAMP). Para reglas propias usa TRANSFORM con ML.IMPUTER, ML.QUANTILE_BUCKETIZE o ML.FEATURE_CROSS: las estadísticas del entrenamiento se reutilizan en ML.PREDICT y evitan el sesgo entre entrenamiento y entrega. ML.TRANSFORM, los modelos TRANSFORM_ONLY, ML.FEATURE_IMPORTANCE y ML.GLOBAL_EXPLAIN sirven para inspeccionar y seleccionar atributos.
- ML.PREDICT puntúa por lotes tablas o consultas dentro de BigQuery sin mover datos, admite THRESHOLD en modelos binarios y aplica automáticamente la cláusula TRANSFORM. ARIMA_PLUS usa ML.FORECAST, ML.EXPLAIN_PREDICT añade atribuciones por fila y ML.RECOMMEND sirve a la factorización de matrices. Para inferencia en línea, registra el modelo en Model Registry y despliégalo en un extremo; los modelos ONNX y remotos también usan ML.PREDICT.
- AutoML divide los datos tabulares en entrenamiento, validación y prueba: aleatoria 80/10/10 por defecto, cronológica con columna de tiempo para datos que cambian con el tiempo o manual para separar, por ejemplo, tiendas. Elige el objetivo según el problema (AUC PR para la clase poco frecuente, log loss obligatorio en multiclase), usa una columna de ponderación para filas importantes y Tabular Workflows cuando necesites controlar cada paso.
- El ajuste supervisado de Gemini desde BigQuery se hace con CREATE MODEL ... REMOTE WITH CONNECTION, OPTIONS(ENDPOINT = ...) y una cláusula AS SELECT con columnas prompt y label (o PROMPT_COL e INPUT_LABEL_COLS). MAX_ITERATIONS se convierte en épocas, DATA_SPLIT_METHOD reserva datos de evaluación para la detención temprana y ML.EVALUATE compara el modelo base con el ajustado. Ajusta cuando las instrucciones no bastan y tienes ejemplos etiquetados.

Building AI solutions using Google Cloud AI APIs or foundational models

- Model Garden reúne modelos fundacionales, ajustables y de tarea específica de Google, socios y abiertos. Usa MaaS para prototipos, tráfico variable y mínima operación; autodespliega para pesos propios, alto volumen predecible, control del hardware o normas contra servicios multiinquilino. Compara candidatos con el Gen AI evaluation service sobre tus datos, limita modelos con una política de organización y revisa los marcados como sospechosos.
- Document AI usa procesadores: Enterprise Document OCR para digitalizar, Form Parser, Layout Parser, Custom Extractor y analizadores preentrenados (facturas, gastos) para extraer, y Custom Classifier o Custom Splitter para clasificar. Vision API ofrece OCR, etiquetas, logotipos, objetos y SafeSearch. Cloud Translation Advanced añade glosarios, traducción adaptativa y traducción de documentos que conserva el formato.
- Empieza con instrucciones y pasa al ajuste solo si no bastan: el ajuste supervisado sirve para formato, clasificación o extracción con cientos de ejemplos; el de preferencias, para pares preferido/rechazado tras un ajuste supervisado; y el refuerzo, para razonamiento verificable. Model Garden ofrece modelos abiertos como API gestionada sin servidores (MaaS) o autodesplegados en tu proyecto; Veo genera vídeo.
- El almacenamiento en caché de contexto reduce coste y latencia con contenido repetido: el implícito es automático (pon el contenido común al principio) y el explícito garantiza el descuento y controla la caducidad. La inferencia por lotes sirve para trabajos no urgentes. Provisioned Throughput reserva capacidad a coste fijo; Priority PayGo da constancia sin compromiso; el extremo global mejora la disponibilidad, pero no garantiza residencia.

2. Collaborating within and across teams to manage data and models

16–16%

Exploring and preprocessing data for ML

- Guarda los datos estructurados y semiestructurados en BigQuery, materializados para entrenar y leídos con la API BigQuery Storage; las imágenes, el vídeo y el audio van en Cloud Storage en contenedores grandes y fragmentados. Las tablas de objetos llevan esos archivos a SQL. AutoML tabular importa de BigQuery o CSV; las imágenes, de JSON Lines o CSV. El texto se personaliza con Gemini.
- Si los datos están en BigQuery y la lógica cabe en SQL, preprocesa allí: suele ser lo más rentable, con tablas particionadas, TRANSFORM en BigQuery ML o BigQuery DataFrames para código estilo pandas a escala. Dataflow sirve para grandes volúmenes, streaming y lógica fuera de SQL, con MLTransform. Managed Service for Apache Spark encaja si ya tienes código Spark, y pandas basta con datos que caben en memoria.
- Feature Store usa tablas o vistas de BigQuery como almacén sin conexión. Los grupos de atributos registran orígenes y columnas, y son obligatorios para series temporales; una vista de atributos puede reunir atributos de varios grupos y se sincroniza de forma programada o continua con un almacén en línea de Bigtable online serving. Los embeddings van a Vector Search, y los recursos deben estar en la región del origen.
- Sensitive Data Protection descubre PII en toda la organización con perfiles de datos, la inspecciona en detalle y la desidentifica (ocultación, enmascaramiento, tokenización, agrupación, desplazamiento de fechas). El hash criptográfico mantiene las uniones sin ser reversible, el análisis de riesgos mide el k-anonimato antes de compartir y el enmascaramiento de BigQuery da acceso parcial por grupo.

Model prototyping using notebooks (e.g., Gemini Enterprise Agent Platform Workbench and Colab Enterprise)

- Colab Enterprise comparte notebooks mediante IAM con usuarios, grupos o dominios (Code Editor, Code Viewer y Notebook Runtime User) y permite desactivar el guardado de resultados, que los demás verían. Workbench controla el acceso con los modos Single user o Service account, opciones de seguridad (root, descarga de archivos), IP externa, CMEK y perímetros de VPC Service Controls.
- Workbench trae JupyterLab con TensorFlow y PyTorch, kernels conda para otras versiones e integración con Managed Service for Apache Spark; Colab Enterprise usa plantillas de entorno con GPU. Hay que ajustar el preprocesamiento solo con el conjunto de entrenamiento (Pipeline de scikit-learn), llamar a model.eval() en PyTorch y usar contenedores personalizados para JAX o dependencias privadas.
- Model Garden reúne modelos fundacionales, ajustables y específicos de Google, socios y Hugging Face; desde la ficha se abren notebooks de ajuste o se despliega con el SDK (OpenModel, list_deploy_options, deploy). Los modelos autodesplegados se ejecutan en tu proyecto y VPC; MaaS es una API gestionada sin servidor. Una política de organización limita los modelos permitidos.

Tracking and running ML experiments

- Agent Platform Pipelines ejecuta sin servidores canalizaciones de Kubeflow Pipelines o de TFX (recomendado con TensorFlow y terabytes de datos). Experiments registra ejecuciones y, con autolog(), guarda parámetros y métricas de scikit-learn o XGBoost; Vertex AI TensorBoard muestra métricas a lo largo del tiempo. Ray on Agent Platform reutiliza código de Ray, y el executor de Workbench programa notebooks.
- Para modelos predictivos, Agent Platform evalúa con precisión, recuperación, AuPRC, F1, matriz de confusión, RMSE o MAE a partir de inferencias por lotes y una verdad fundamental, también como componente de canalización; ML.EVALUATE admite un umbral propio. Para IA generativa, el Gen AI evaluation service usa rúbricas adaptativas (GENERAL_QUALITY), estáticas (GROUNDING, SAFETY), métricas de cálculo y un modelo juez configurable.
- Agent Platform Experiments registra parámetros, métricas de resumen y métricas de series temporales (guardadas en Vertex AI TensorBoard) y compara ejecuciones, también de canalizaciones. Agent Platform ML Metadata guarda el linaje de artefactos y ejecuciones por proyecto y región; Knowledge Catalog da linaje de toda la organización. Model Registry gestiona versiones y alias mutables.

3. Scaling prototypes into ML models

21–21%

Building models given the task considering cost, complexity, latency, and scalability

- El tipo de modelo sale de la tarea: LOGISTIC_REG o árboles potenciados para clasificar, KMEANS para segmentar sin etiquetas, MATRIX_FACTORIZATION (IMPLICIT sin puntuaciones) para recomendar, ARIMA_PLUS o ARIMA_PLUS_XREG (con covariables) para series temporales, wide-and-deep o autoencoder para datos dispersos, y un LLM como Gemini (AI.CLASSIFY) cuando no hay etiquetas.
- AutoML no requiere código y sirve para prototipos rápidos con objetivos predefinidos; BigQuery ML entrena con SQL sin sacar los datos del almacén; el entrenamiento sin servidores admite cualquier framework, pérdida o métrica; los training clusters reservan capacidad garantizada. Agent Platform Pipelines orquesta flujos reproducibles sin servidores, y Tabular Workflows da control paso a paso sobre AutoML.
- La inferencia en línea exige desplegar el modelo en un extremo; la inferencia por lotes es asíncrona sobre un modelo sin desplegar y es la única opción para la previsión tabular de AutoML. La división del tráfico permite sustituir un modelo de forma gradual. El extremo público dedicado aísla el tráfico y admite streaming; el privado con Private Service Connect admite VPC Service Controls y división del tráfico.
- BigQuery Explainable AI ofrece explicaciones locales con ML.EXPLAIN_PREDICT (regresión lineal y logística, árboles potenciados, bosques aleatorios) y globales con ML.GLOBAL_EXPLAIN (requiere ENABLE_GLOBAL_EXPLAIN), además de ML.FEATURE_IMPORTANCE, ML.ADVANCED_WEIGHTS con valores p y ML.EXPLAIN_FORECAST para ARIMA_PLUS. Vertex Explainable AI está obsoleto: Google sugiere SHAP o LIME.

Training models

- Los datos no estructurados (imágenes, audio, vídeo) y formatos como TFRecord van a Cloud Storage y se leen con Cloud Storage FUSE en archivos grandes y pocos; las tablas de objetos de BigQuery los exponen a SQL. Los datos tabulares van mejor en BigQuery si son grandes o anidados. AutoML usa archivos JSON Lines o CSV con ml_use, y el ajuste de Gemini usa JSON Lines en Cloud Storage.
- Los datos no estructurados (imágenes, texto, TFRecord) suelen ir en Cloud Storage, que el entrenamiento sin servidores monta en /gcs/ con Cloud Storage FUSE; un recurso compartido NFS da baja latencia y semántica POSIX, y BigQuery sirve para datos estructurados. Los conjuntos de datos gestionados reparten los datos y pasan las URI con AIP_TRAINING_DATA_URI. Si los datos no caben en memoria, se leen en streaming.
- El entrenamiento sin servidores ejecuta código de cualquier framework en contenedores prediseñados (script o distribución de Python, poco esfuerzo) o personalizados (otros lenguajes o dependencias privadas). Una canalización de entrenamiento puede registrar el modelo en Model Registry. Tabular Workflows da control por pasos sobre AutoML; los recursos persistentes acortan el arranque; Kubeflow Trainer en GKE es portable.
- Agent Platform ejecuta el código en proyectos gestionados por Google: indica el proyecto de forma explícita (CLOUD_ML_PROJECT_ID) para evitar errores 403. Ante falta de memoria, optimiza el uso o elige una máquina mayor; ante reinicios, guarda puntos de control al menos cada cuatro horas. La shell interactiva y Cloud Logging ayudan a depurar, y unas métricas excelentes que fallan en producción suelen indicar filtración de datos.
- Un trabajo de ajuste de hiperparámetros ejecuta varias pruebas del código, que debe leer los hiperparámetros como argumentos de línea de comandos e informar de la métrica con cloudml-hypertune. Por defecto usa optimización bayesiana; la búsqueda en cuadrícula exige parámetros enteros, categóricos o discretos. Las pruebas en paralelo acortan el tiempo pero restan eficacia, y Vizier solo sugiere valores para sistemas que evalúas tú.

Choosing appropriate hardware for training

- Las CPU sirven para prototipos y modelos sencillos; las GPU, para modelos medianos o grandes con operaciones personalizadas de PyTorch o JAX; las TPU, para modelos dominados por matrices, sin operaciones personalizadas en el bucle, con lotes grandes y entrenamientos largos, pero no para formas dinámicas ni alta precisión. Spot VMs abarata trabajos tolerantes a fallos y Dynamic Workload Scheduler espera a GPU escasas.
- El entrenamiento distribuido sin servidores usa grupos de trabajadores (réplica principal en workerPoolSpecs[0], trabajadores, servidores de parámetros, evaluadores) descritos en CLUSTER_SPEC o TF_CONFIG. El paralelismo de datos replica el modelo con DistributedDataParallel; si no cabe, FSDP lo fragmenta o el paralelismo de modelos lo reparte. Las TPU encajan con modelos matriciales grandes; con TPU VM hay un solo grupo y una réplica.

4. Serving and scaling models

20–20%

Serving models

- La inferencia en línea es síncrona contra un modelo desplegado en un extremo; la inferencia por lotes es un BatchPredictionJob asíncrono sobre un Model sin desplegar, con entrada y salida en BigQuery o Cloud Storage, que no escala automáticamente. Model Garden ofrece modelos abiertos como MaaS sin servidores o autodesplegados en tu proyecto y VPC. Cloud Run con GPU es gestionado y baja a cero; GKE sirve servidores como vLLM con Kubernetes.
- Los contenedores prediseñados de inferencia (TensorFlow, PyTorch, scikit-learn y XGBoost) sirven con mínima configuración si el artefacto cumple su formato: SavedModel, model.mar con controlador, model.joblib o model.bst. Para otros frameworks, dependencias propias o lógica extra se usa un contenedor personalizado en Artifact Registry con un servidor HTTP en 0.0.0.0 en el puerto de containerSpec.ports, que lee los artefactos de AIP_STORAGE_URI.
- Model Registry es el repositorio central del ciclo de vida de los modelos personalizados, AutoML y BigQuery ML (registrados sin exportar con MODEL_REGISTRY y VERTEX_AI_MODEL_ID). Cada reentrenamiento se importa como versión nueva; un alias es una referencia mutable y única a una versión, el alias default marca la versión usada si no se indica otra y las etiquetas organizan. Copy model lleva un modelo a otra región o proyecto.
- En un despliegue canario o una prueba A/B se despliegan varios modelos en el mismo extremo y se reparte el tráfico con porcentajes que suman 100. La aplicación mantiene la URL, revertir es cambiar la división y el campo deployedModelId de cada respuesta indica qué modelo respondió. Un despliegue gradual sustituye el modelo y migra todo el tráfico; los extremos con acceso privado a servicios no admiten divisiones.
- Las rutinas de inferencia personalizadas crean contenedores con un Predictor (load, preprocess, predict, postprocess) sin escribir servidor ni Dockerfile, y un Handler para solicitudes no JSON. En lotes, instanceConfig excluye o incluye campos sin código. La cláusula TRANSFORM de BigQuery ML aplica el preprocesamiento en ML.PREDICT, Dataflow RunInference encadena pasos en streaming y un response_schema fija la salida de Gemini.

Scaling online model serving

- Feature Store usa tablas o vistas de BigQuery como almacén sin conexión y sirve los valores más recientes desde vistas de atributos en un almacén en línea; Bigtable online serving es el tipo vigente, admite CMEK y sincronización programada o continua (esta última exige grupos de atributos). Optimized online serving está obsoleto y los embeddings van a Vector Search.
- Para inferencia en línea hay extremos públicos dedicados (recomendados, con aislamiento), públicos compartidos, privados dedicados con Private Service Connect (recomendados para producción privada, varios proyectos consumidores sin emparejamiento) y privados con acceso privado a servicios, que solo admiten un modelo por extremo y no dividen el tráfico. En los públicos y en PSC se puede desplegar un modelo nuevo con poco tráfico e ir subiéndolo.
- Las CPU sirven para prototipos y modelos pequeños; las GPU, para modelos con operaciones personalizadas que corren en parte en CPU, y solo se usan con SavedModels de TensorFlow o contenedores preparados (no scikit-learn ni XGBoost); las TPU, para modelos dominados por matrices con formas estáticas y lotes grandes. Conviene medir varios tipos de máquina, y las Spot VMs abaratan trabajos tolerantes a interrupciones.
- Un modelo desplegado escala si maxReplicaCount supera a minReplicaCount; autoscalingMetricSpecs permite escalar por CPU, GPU o número de peticiones, y la cuota se reserva según maxReplicaCount. La inferencia por lotes no escala sola. Muchos modelos con poco tráfico pueden compartir VM en un DeploymentResourcePool, y para Gemini se elige entre Standard PayGo, Priority PayGo, Provisioned Throughput y lotes, mejor por el extremo global.
- El entorno de ejecución optimizado de TensorFlow abarata y acelera la inferencia sin cambiar código, con precompilación XLA, compresión (cuantización, poda) y partes de modelos tabulares en GPU. En Gemini, el ajuste supervisado reduce latencia y coste al acortar las instrucciones, y el ajuste eficiente de parámetros cuesta menos que el completo. El ajuste de hiperparámetros pasa valores por argumentos y recibe métricas con cloudml-hypertune.

5. Automating and orchestrating ML pipelines

18–18%

Developing end-to-end ML pipelines

- En una canalización de entrenamiento continuo, la validación de datos va antes de entrenar: un sesgo de esquema detiene la canalización y un cambio estadístico justifica reentrenar. La validación del modelo compara métricas de prueba con el modelo actual, revisa segmentos y prueba el despliegue; puede automatizarse con componentes de evaluación y dsl.If. En BigQuery ML, ML.VALIDATE_DATA_SKEW usa las estadísticas de entrenamiento guardadas.
- Agent Platform Pipelines ejecuta sin servidores canalizaciones de Kubeflow Pipelines o TFX compiladas a YAML, usa Google Cloud Pipeline Components prediseñados y plantillas versionadas en Artifact Registry, y guarda el linaje en ML Metadata. Managed Airflow es el orquestador recomendado cuando el flujo abarca sistemas heterogéneos, y Ray en Agent Platform ejecuta código de Ray existente con cambios mínimos.
- El sesgo entre el entrenamiento y la entrega aparece cuando el preprocesamiento se duplica fuera del modelo. La cláusula TRANSFORM de BigQuery ML se aplica sola en ML.PREDICT (lo hecho en SELECT no se exporta), MLTransform reutiliza artefactos con write_artifact_location y read_artifact_location, y Feature Store, las rutinas de inferencia personalizadas o un componente compartido mantienen una sola implementación.

Automating model retraining

- La política de reentrenamiento depende de cómo llegan las etiquetas y de lo caro que sea entrenar: programación con cron cuando los datos llegan de forma sistemática, activación por evento cuando llegan de forma puntual, evaluación continua cuando hay etiquetas rápidas y alertas de desvío de Model Monitoring cuando no las hay. La validación de datos detiene la canalización ante sesgos de esquema.
- Con datos nuevos se ejecuta la canalización de CT ya desplegada; con código nuevo, la CI/CD despliega una nueva canalización. Los activadores de Cloud Build ejecutan cloudbuild.yaml para probar, crear imágenes etiquetadas con $COMMIT_SHA, subirlas a Artifact Registry y compilar la canalización, con archivos ignorados y aprobación obligatoria; los alias de Model Registry promocionan versiones.

6. Monitoring AI solutions

13–13%

Identifying risks to AI solutions

- Model Armor analiza instrucciones y respuestas con detección de inyección de instrucciones y jailbreak, URL maliciosas, filtros de IA responsable y Sensitive Data Protection, en modo Inspect only o Inspect and block, con plantillas o floor settings. Los filtros de Gemini cubren categorías de daño, una expresión regular no capta paráfrasis y VPC Service Controls impide que datos y modelos salgan del perímetro.
- Agent Platform mide la equidad por segmentos: las métricas de sesgo en los datos (Difference in Population Size, Difference in Positive Proportions in True Labels) se calculan antes de entrenar, y las del modelo (Accuracy, Recall o Specificity Difference) después, con DetectDataBiasOp y DetectModelBiasOp en canalizaciones. En IA generativa se depuran los datos de ajuste, se prueba con usuarios representativos y se mantiene supervisión humana.
- La explicabilidad local atribuye cada predicción a sus atributos respecto a una predicción de referencia; la global agrega esas atribuciones para todo el modelo. En BigQuery ML se usan ML.EXPLAIN_PREDICT (local), ML.GLOBAL_EXPLAIN (requiere ENABLE_GLOBAL_EXPLAIN), ML.FEATURE_IMPORTANCE (árboles), ML.ADVANCED_WEIGHTS (p-valores) y ML.EXPLAIN_FORECAST (ARIMA_PLUS). Vertex Explainable AI está obsoleto: Google propone SHAP o LIME.

Monitoring, testing, and troubleshooting AI solutions

- Model Monitoring v1 se configura en un extremo y comparte tipo de detección, frecuencia y muestreo entre sus modelos; v2 asocia un model monitor a una versión de Model Registry, sirve también fuera de Agent Platform y admite ejecuciones programadas con ventanas y desfases. Las alertas llegan por correo, canales de Cloud Monitoring o Cloud Logging. La evaluación continua compara la inferencia con etiquetas reales nuevas.
- El sesgo entre el entrenamiento y la entrega compara producción con los datos de entrenamiento y exige tenerlos; el desvío compara producción con producción anterior. El desvío de atribución de atributos sigue la contribución de cada atributo (SHAP) y ayuda a detectar la deriva de conceptos. Las distancias son L-infinito para categóricos y Jensen-Shannon para numéricos; en lotes, solo sesgo al crear el trabajo.
- El Gen AI evaluation service evalúa con rúbricas adaptativas (recomendadas), rúbricas estáticas, métricas basadas en cálculo como ROUGE cuando hay verdad fundamental y funciones personalizadas; el modelo juez se valida con valoraciones humanas y se ajusta con inversión de respuestas o multimuestreo. Los agentes se evalúan por trayectoria y respuesta final, y los modelos MaaS se vigilan con el panel de observabilidad y registros en BigQuery.