Microsoft DP-100 Azure Data Scientist Certification Study Notes

Code: DP-100

Azure ML Workspace

Azure Machine Learning Workspace

Central resource for ML development

Core Components

Component	Purpose	Key Features
Workspace	Top-level resource	Contains all ML assets
Compute Instances	Development VMs	Jupyter, VS Code integration
Compute Clusters	Training resources	Auto-scaling, low-priority VMs
Datastores	Data connections	Azure Storage, ADLS, SQL
Data Assets	Versioned datasets	URIs, MLTable, file references
Environments	Runtime configs	Docker images, conda specs

Associated Resources

Storage Account

Default datastore, artifacts, logs

Key Vault

Secrets, connection string

Container Registry

Docker images for environments

Application Insights

Monitoring, telemetry, logs

Compute Types

- **Compute Instance:** Single VM for development and testing
- **Compute Cluster:** Multi-node for distributed training
- **Kubernetes:** AKS for inference workloads
- **Serverless:** On-demand compute for training jobs
- **Attached:** External compute (HDInsight, Databricks)

Exam Focus Areas

- Compute clusters auto-scale to 0 nodes when idle
- Use low-priority VMs for cost savings (may be preempted)
- Datastores abstract connection details from code
- Data assets enable versioning and lineage tracking

Data Exploration

Data Exploration & Preparation

Working with data in Azure ML

Data Asset Types

Type	Use Case	Format
URI File	Single files	uri_file
URI Folder	Directories	uri_folder

Type	Use Case	Format
MLTable	Tabular data	mltable (with MLTable file)

Data Access Patterns

Download

Copy data to local compute

Mount

Mount as file system (read/write)

Direct

Stream data directly

Feature Engineering

- **Normalization:** Scale features to common range (MinMax, StandardScaler)
- **Encoding:** Convert categorical to numeric (OneHot, Label)
- **Imputation:** Handle missing values (mean, median, mode)
- **Feature Selection:** Choose relevant features (correlation, importance)
- **Binning:** Group continuous values into categories

MLTable

MLTable allows schema definition and transformations in a YAML file. It's the preferred format for tabular datasets used in AutoML and pipelines.

Exam Focus Areas

- Use MLTable for AutoML jobs and tabular data
- Mount is efficient for large datasets (no full download)
- Version data assets for reproducibility

- Datastores use service principal or managed identity

Model Training

Model Training in Azure ML

Training approaches and techniques

Training Options

Method	Use Case	Features
AutoML	Automated model selection	Algorithm search, hyperparameter tuning
Designer	Visual ML pipeline	Drag-drop, no code
Python SDK	Custom training scripts	Full control, flexibility
CLI v2	Command-line workflows	YAML-based configuration

AutoML Task Types

Classification

Predict categories (binary, multi-class)

Regression

Predict continuous values

Time Series

Forecast future values

Computer Vision

Image classification, object detection

NLP

Text classification, NER

Hyperparameter Tuning

- **Grid Search:** Exhaustive search over parameter grid
- **Random Search:** Random sampling of parameter space
- **Bayesian:** Smart sampling based on prior results
- **Early Termination:** Stop poor-performing runs (Bandit, Median, Truncation)

Distributed Training

Data Parallelism

Split data across nodes

Model Parallelism

Split model across nodes

PyTorch DDP

DistributedDataParallel

Horovod

Framework-agnostic distribution

Exam Focus Areas

- AutoML handles featurization, algorithm selection, tuning
- Use Bayesian sampling for efficient hyperparameter search
- Early termination saves compute costs on poor runs
- MLflow logs metrics, parameters, and artifacts automatically

MLOps & Pipelines

Azure ML Pipelines & MLOps

Operationalize ML workflows

Pipeline Components

Component	Description	Use
Command	Run scripts	Training, preprocessing
Parallel	Batch processing	Large-scale inference
Pipeline	Multi-step workflow	Chain components
Sweep	Hyperparameter tuning	Find best parameters

MLflow Integration

Tracking

Log metrics, params, artifacts

Models

Model registry with versioning

Projects

Reproducible runs

Serving

Deploy models as endpoints

CI/CD for ML

- **Azure DevOps:** Pipelines for training and deployment
- **GitHub Actions:** Workflow automation
- **Model Registry:** Version and stage models (staging → production)
- **Endpoints:** Blue-green deployments, traffic splitting

> **Responsible AI**
>
> Use the Responsible AI dashboard to understand model behavior: Error Analysis, Fairness, Interpretability, and Counterfactuals.

Exam Focus Areas

- Pipelines automate data prep → training → deployment
- MLflow is the default tracking framework in Azure ML
- Register models to enable versioning and deployment
- Use managed endpoints for simplified deployment

Deployment

Model Deployment

Deploy models to production

Endpoint Types

Type	Use Case	Features
Online Managed	Real-time inference	Auto-scaling, load balancing
Online Kubernetes	AKS deployment	More control, custom infra
Batch	Large-scale batch	Scheduled, parallel processing
Serverless	On-demand inference	Pay-per-use, no idle cost

Deployment Strategies

Blue-Green

Two deployments, switch traffic

Canary

Gradual traffic shift

A/B Testing

Compare model versions

Shadow

Parallel deployment, compare results

Scoring Script

- **init():** Load model, one-time setup
- **run(data):** Process each request, return predictions
- **Model path:** Use AZUREML_MODEL_DIR environment variable
- **Dependencies:** Specify in environment or conda file

Monitoring & Retraining

Data Drift

Detect input distribution changes

Model Drift

Prediction quality degradation

Alerts

Trigger retraining pipelines

A/B Testing

Compare model versions

Exam Focus Areas

- Managed online endpoints handle scaling automatically
- Use traffic allocation for blue-green deployments
- Batch endpoints for processing large datasets
- Monitor data drift to know when to retrain

Responsible AI

Responsible AI Practices

Build trustworthy ML solutions

Responsible AI Principles

Fairness

Equal treatment across groups

Reliability

Consistent, expected behavior

Privacy

Protect sensitive data

Inclusiveness

Accessible to all users

Transparency

Explainable decisions

Accountability

Human oversight

Responsible AI Dashboard

Component	Purpose	Insights
Error Analysis	Find failure modes	Error cohorts, patterns
Fairness	Assess bias	Metrics across groups
Interpretability	Explain predictions	Feature importance, SHAP
Counterfactuals	What-if scenarios	Change inputs, see effects
Causal	Cause-effect analysis	Treatment effects

Model Interpretability

- **Global:** Overall feature importance across dataset
- **Local:** Explanation for individual predictions
- **SHAP:** SHapley Additive exPlanations values
- **LIME:** Local Interpretable Model-agnostic Explanations

- **InterpretML:** Microsoft's interpretability package

⚠ Exam Focus Areas

- Use Responsible AI dashboard to debug models
- Error Analysis helps find where model fails most
- Fairness metrics reveal bias across demographic groups
- Counterfactuals show what changes would flip predictions