
Index

A

- ACI (Azure Container Instances), 369
- AKS (Azure Kubernetes Service), 370
- alerts, defined, 369
- Amazon ECR, defined, 369
- Amazon EKS, defined, 369
- Amazon Mechanical Turk, 351
- antipatterns, 357-362
- API services, authentication, 240
- App Engine (see Google App Engine)
- App Runner, 202, 221-222
- app.py, 210
- Apple
 - Core ML (see Core ML)
 - Create ML, 132-136
 - mobile development ecosystem, 131-139
- ASICS, 19
- authentication
 - API services, 240
 - authenticating microservices to cloud functions, 338-340
 - Azure, 238-240
 - key-based, 240
 - service principal, 238-240
- automated checks, 113
- Automator's law, 30, 118
- AutoML, 31, 117-159
 - Apple's ecosystem and, 131-139
 - AWS, 146-150
 - Azure AutoML, 144-146
 - basics, 118-129
 - Create ML, 132-136
 - data science versus, 123
 - Feature Stores, 127-129
 - FLAML, 153-154
 - Google's AutoML and edge computer vision, 136-139
 - Kaizen versus KaizenML, 125-127
 - Ludwig, 151
 - MLOps industrial revolution, 123-125
 - model explainability, 154-158
 - open source solutions, 151-154
- autoscaling, defined, 369
- AWS, 187-234
 - AutoML, 146-150
 - AWS Lambda recipes, 223-229
 - basics, 188-209
 - best practices for bootstrapping MLOps capabilities, 233
 - CaaS, 198-203
 - career advice with AWS ML evangelist Julien Simon, 231-233
 - certification options, 375-390
 - Certified Machine Learning specialist certification, 2
 - CLI tools for MLOps Cookbook project, 211-218
 - computer vision, 204-205
 - DataOps tools, 15
 - Flask microservice, 218-222
 - getting started with, 189-205
 - Hugo static S3 websites, 192-194
 - MLOps Cookbook on, 209-222
 - MLOps on, 206-209
 - no-code/low-code AWS Comprehend solution, 190
 - real-world ML applications, 229-233
 - serverless cookbook, 194-198

- Sports Social Network case study, 229-231
 - AWS App Runner, 202, 221-222
 - AWS Certified Machine Learning - Specialty certification, 381-390
 - data engineering, 382-387
 - exploratory data analysis, 387-389
 - MLOps, 390
 - AWS Cloud Practitioner, certification test questions for, 375-381
 - AWS Cloud9, 25, 193, 369
 - AWS Cloudshell, 24-25
 - AWS CloudWatch, 163
 - AWS Comprehend, 190
 - AWS DeepLens, 204-205
 - AWS DeepRacer, 52
 - AWS Lambda, 33
 - defined, 369
 - deploying with SAM, 223-229
 - recipes, 223-229
 - SAM containerized deploy, 224-229
 - SAM Local, 223
 - serverless applications, 194-198
 - AWS S3
 - Hugo static S3 websites, 192-194
 - SageMaker and, 176-181
 - AWS SageMaker (see SageMaker)
 - AWS Serverless Application Model (see SAM)
 - AWS Solutions Architect certification test questions, 375-381
 - Azure, 235-263
 - Application Insights, 253
 - authenticating API services, 240
 - authentication, 238-240
 - Azure CLI and Python SDK, 236-238
 - Azure Machine Learning designer, 260-262
 - compute instances, 240-242
 - debugging locally, 254-257
 - deploying, 242-245
 - deploying a model, 248-251
 - deploying models to a compute cluster, 246-251
 - deploying ONNX to, 307-310
 - Microsoft Certified: Azure Data Scientist certification, 2
 - ML lifecycle, 262
 - ML pipelines, 257-262
 - MLOps for, 235-263
 - publishing pipelines, 259
 - registering models, 243-244
 - retrieving logs, 252
 - service principal, 238-240
 - troubleshooting deployment issues, 251-257
 - versioning datasets, 245
 - Azure AI Engineer Associate certification, 391
 - Azure Associate Data Scientist certification, 391
 - Azure AutoML, 144-146
 - Azure Container Instances (ACI), 369
 - Azure Kubernetes Service (AKS), 370
 - Azure ML, monitoring drift with, 182-184
 - Azure Percept, 84
 - Azure Pipelines, 56-63
- ## B
- Baseball_Predictions_Export_Model.ipynb, 211
 - baseline dataset, target dataset versus, 179
 - Bash shell, 26-29
 - configuration, 27
 - files and navigation, 27
 - input/output, 27
 - list files, 26
 - run commands, 26
 - writing a script, 28
 - BigQuery (see Google BigQuery)
 - bind mount, 87
 - black tool (Python), 370
 - Blake, William, 408
 - blue-green deployment, 111
 - Bogen, Joseph, xiii, 1, 23, 67, 93, 117, 161, 187, 235, 265, 293, 317, 347
 - build server, defined, 370
- ## C
- CaaS (see container as a service)
 - canary deployment, 111
 - career planning
 - pear (PPEAR) revenue strategy, 399-402
 - thinking like a venture capitalist, 399
 - case studies, 347-367
 - critical challenges in MLOps, 357-362
 - ethical/unintended consequences, 358
 - Francesca Lazzeri interview, 362
 - intermittent fasting, 409-414
 - MLOps projects as Sqor sports social network, 350-355
 - perfect technique versus the real world, 355-357
 - Piero Molino interview, 360-362

- unlikely benefits of ignorance in building
 - ML models, 348-350
- CD (see continuous delivery (CD))
- certifications, 375-392
 - AWS Certified Machine Learning - Specialty, 381-390
 - AWS Cloud Practitioner, 375-381
 - AWS Solutions Architect, 375-381
 - Azure Data Scientist/AI Engineer, 391
 - SQL-related, 391
- challenges in MLOps
 - ethical/unintended consequences, 358
 - focus on prediction accuracy versus the big picture, 359
 - lack of operational excellence, 358
- Charpentier, Emmanuelle, 121
- CI/CD (continuous integration/continuous delivery)
 - implementation of, 7
 - parallels to recovery from sports injuries, 94
- CI/CD pipeline, quality-control checks for, 32
- CircleCI, defined, 370
- CLI tools (see command line interface tools)
- cli.py, 210
- cloud computing
 - foundations/building blocks, 29-31
 - getting started, 31-32
 - machine learning and, 3
- cloud MLOps, observability for, 163
- cloud native applications, defined, 370
- cloud pipelines
 - continuous delivery, 107-115
 - controlled rollout of models, 110
 - testing techniques for model deployment, 112-115
- Cloud Run, 268
- cloud shell development environments, 24-26
- Cloud9 (see AWS Cloud9)
- Cloudshell, 24-25
- cluster
 - configuring, 246-248
 - deploying models to a compute cluster, 246-251
 - inference versus compute, 247
- command line interface (CLI) tools, 317-344
 - basics, 321-331
 - building a cloud-based CLI, 341
 - creating a dataset linter, 321-328
 - machine learning CLI workflows, 342-344
 - MLOps Cookbook project, 211-218
 - modularizing, 328-331
 - Python packaging, 319
 - requirements file, 320
- command line, Linux, 23
- Common Vulnerabilities and Exposures (CVEs), 75
- competitive advantage, 190
- compute cluster
 - deploying models to, 246-251
 - inference cluster versus, 247
- Compute Engine, 267
- compute instances, Azure, 240-242
- computer vision
 - AWS, 204-205
 - Google's AutoML and edge computer vision, 136-139
- configuration
 - Bash shell, 27
 - cluster, 246-248
 - configuring continuous integration with GitHub Actions, 13-14
 - runtime, 259
- container as a service (CaaS)
 - AWS, 198-203
 - MLOps design pattern, 366
- containerized workflow, 224-229
- containers, 68-80
 - best practices, 74-76
 - build once, run many MLOps workflow, 91
 - creating, 69-71
 - defined, 370
 - managed ML systems and, 89-91
 - monetizing MLOps, 90
 - running, 72-73
 - runtime, 69
 - serving a trained model over HTTP, 76-80
- continuous delivery (CD), 6, 93-115
 - (see also CI/CD)
 - cloud pipelines for, 107-115
 - defined, 370
 - DevOps best practice, 5
 - GCP and, 270-272
 - infrastructure as code for continuous delivery of ML models, 99-106
 - packaging for ML models, 95-99
 - testing techniques for model deployment, 112-115
- continuous improvement, 8, 114

- continuous integration (CI), 6
 - (see also CI/CD)
 - configuring with GitHub Actions, 13-14
 - defined, 370
 - DevOps best practice, 5
 - GCP and, 270-272
- convergence, 49
- Coral Project, 81-84
- Core ML, 132, 136-139, 310-314
- Create ML, 132-136
- customer-focused education, 422
- CVEs (Common Vulnerabilities and Exposures), 75
- cybersecurity, 365

D

- data drift (see drift)
- data engineering
 - AWS Certified Machine Learning - Specialty certification, 382-387
 - defined, 370
 - GCP, 282-286
 - ML hierarchy of needs and, 15
- data governance, 365, 385
- data lake, 15, 382
- data science
 - AutoML versus, 123
 - as foundational skill, 54-56
- data warehouses, Feature Stores versus, 129
- DataOps
 - GCP, 282-286
 - ML hierarchy of needs and, 15
- dataset linter, 321-328
- datasets
 - baseline versus target, 179
 - versioning, 245
- Davis, Purnell, 352
- debugging (see troubleshooting)
- deep learning intuition, 49-49
- DeepLens, 204-205
- dependencies
 - defining in requirements.txt file, 320
 - pinning, 74
 - Python packaging and, 319
- deployment
 - Azure, 242-245
 - blue-green versus canary, 111
 - deploying a model, 248-251
 - MLOps, 19

- registration, 243-244
- troubleshooting deployment issues, 251-257
- descriptive statistics, 37-41, 39
- design patterns, 366
- development environments, 24-26
- DevOps
 - best practices, 5
 - defined, 5
 - implementing, 8-13
 - MLOps and, 5-7
 - MLOps feedback loop, 18
- diet, intermittent fasting and, 409-414
- disaster recovery, defined, 370
- disruption (see education disruption)
- Docker
 - containers, 69
 - defined, 371
 - Dockerfile, 69, 211
 - format container, 370
- Doudna, Jennifer, 121
- drift
 - monitoring with AWS SageMaker, 175-182
 - monitoring with Azure ML, 182-184

E

- EDA (exploratory data analysis), 39, 387-389
- edge devices, 80-89
 - Apple's ecosystem, 131-139
 - ASICS and, 19
 - Azure Percept, 84
 - Coral, 81-84
 - ORT model format and, 314-315
 - porting over non-TPU models, 86-89
 - TFHub, 85
- education disruption, 418-425
 - current state of higher education that will be disrupted, 420-421
 - 10X better education, 421-425
- educational resources, 415-426
 - additional MLOps critical thinking questions, 415-418
 - additional MLOps educational materials, 418
 - education disruption, 418-425
- Elastic Beanstalk Flask app, 207-209
- ELI5, 155, 157
- engineering, science versus, 2
- explainability, 145, 154-158
- exploratory data analysis (EDA), 39, 387-389

F

- FaaS (see function as a service)
- Faber, Issac, 359
- Fargate, 198-203
- fasting, intermittent, 409-414
- Feature Stores, 127-129
- feedback loop, 18
- FLAML, 153-154
- Flask microservice, 218-222
 - automatically building container via GitHub Actions, 220
 - AWS App Runner and, 221-222
 - build example, 199-202
 - containerized, 219
- food, intermittent fasting and, 409-414
- foundational skills for MLOps, 23-63
 - Bash shell and commands, 26-29
 - building an MLOps pipeline from zero, 56-63
 - cloud computing foundations/building blocks, 29-31
 - cloud shell development environments, 24-26
 - doing data science, 54-56
 - getting started with cloud computing, 31-32
 - Linux command line, 23
 - machine learning key concepts, 50-53
 - math for programmers crash course, 37-49
 - Python (crash course), 33-35
 - Python (tutorial), 36
- Fox, Justin, 119
- function as a service (FaaS)
 - AWS Lambda, 33
 - defined, 371

G

- GCP (see Google Cloud Platform)
- Gilovich, Thomas, 118
- GitHub Actions
 - automatically building container via, 220
 - configuring continuous integration with, 13-14
 - GCP versus, 271
- GitHub Container Registry, pushing container to, 220
- GitHub, file size limitations for, 100
- GKE (Google Kubernetes Engine), 268, 371
- Google
 - AutoML and edge computer vision, 136-139

- Professional Machine Learning Engineer certification, 2

- Google App Engine, 268, 287
- Google BigQuery, 268, 280-281
- Google Cloud Build, 270
- Google Cloud Functions, 282-286
- Google Cloud Platform (GCP), 265-290
 - advantages of using, 267
 - applied data engineering on, 282-286
 - certifications, 391
 - cloud native database choice/design, 280-281
 - continuous integration/continuous delivery, 270-272
 - disadvantages of using, 265-266
 - Kubernetes and, 272-280
 - major components, 267
 - operationalizing ML models, 287-289
 - overview, 265-281
- Google Cloud Run, 268
- Google Kubernetes Engine (GKE), 268, 371
- greedy algorithms, 42-48

H

- Haber, Jonathan, 356
- Hardgrove, Jacob, 357
- Hargadon, Andrew, 367
- Harris, Tristan, 358
- health issues, in home work area, 395
- hierarchy of needs, ML, 7-19
 - configuring continuous integration with GitHub Actions, 13-14
 - DataOps and data engineering, 15
 - implementing DevOps, 8-13
 - MLOps at top of, 17-19
 - platform automation, 16
- hiring process, disruption of, 425
- hobbies, 408
- Horizontal Pod Autoscaler (HPA), 275
- housing affordability, 424
- HTTP, serving a trained model over, 76-80
- htwtmlb.csv, 210
- Hugo websites, 192-194

I

- IaC (see infrastructure as code)
- ignorance, benefits of, 348-350
- implementation of MLOps
 - data governance and cybersecurity, 365

- design patterns, 366
- global recommendations, 364
- recommendations for, 364-366

inference clusters, 247

infrastructure as code (IaC)

- continuous delivery of ML models and, 99-106
- DevOps best practice, 6

input/output

- Bash shell, 27

instructions (container keywords), 70

instrumentation (DevOps best practice), 6

intermittent fasting, 409-414

interoperability, 293-315

- Apple Core ML, 310-314
- edge integration, 314-315
- importance of, 294-296
- ONNX, 296-310

IPython interpreter, defined, 371

Isaacson, Walter, 121

J

James, LeBron, 352

job markets, regional, 424

JSON, defined, 371

K

Kaggle, 206

Kaizen, 4, 125-127

KaizenML, 125-127, 127-129

kernel density plot, 39

key terms, 369-373

key-based authentication, 240

Koonin, Steven, 41, 360

Kubernetes, 272-280

- blue-green deployment, 111
- defined, 371
- Horizontal Pod Autoscaler, 275

Kubernetes clusters, 371

Kubernetes containers, 371

Kubernetes pods, 371

Kubernetes-centric design, 366

L

Lazzeri, Francesca, 362

life-long learning, 423

lifecycle, Azure and, 262

linter

- containers and, 74
- creating a dataset linter, 321-328
- modularizing, 328-331

linting, 32, 114

Linux command line, 23

load testing, defined, 371

Locust, defined, 371

logging, 164

- (see also monitoring)
- basics, 164
- defined, 371
- modifying log levels, 169
- Python, 165-172
- retrieving logs, 252

Loranger, Rob, 229-231

Loukides, Mike, 367

ls command, 26

Ludwig AutoML, 151

M

machine learning (generally)

- cloud computing and, 3
- interoperability (see interoperability)
- key concepts, 50-53

machine learning engineering, tools and processes used in, 3

magical thinking, 119

Makefile, 9, 210

- defined, 372
- reasons to use, 10

managed ML systems, containers for, 89-91

Maslow's hierarchy of needs, 7

math for programmers

- crash course, 37-49
- descriptive statistics/normal distributions, 37-41
- optimization, 41-49

Mechanical Turk, 351

metrics

- defined, 372
- for model monitoring, 174

microservices, 331-342

- authenticating to cloud functions, 338-340
- building a cloud-based CLI, 341
- creating a serverless function, 333-338
- defined, 69, 372
- as DevOps best practice, 5

Microsoft (see Azure entries)

migrate (term), 372

- mistake mindset, 407
- ML engineering (see hierarchy of needs, ML)
- mllib.py, 210
- MLOps (generally)
 - basics, 1-21
 - defined, 4
 - deployment possibilities, 19
 - DevOps and, 5-7
 - feedback loop, 18
 - foundational skills for (see foundational skills)
 - global recommendations, 364
 - hierarchy of needs, 7-19
 - Kaizen versus KaizenML, 125-127
 - recommendations for implementing, 364-366
 - rise of machine learning engineers and MLOps, 2
 - technical portfolio for, 403-408
 - at top of ML hierarchy of needs, 17-19
- MLOps Cookbook
 - AWS and, 209-222
 - CLI tools, 211-218
 - Flask microservice, 218-222
- MLOps industrial revolution, 123-125
- MLOps platform, 366
- mobile phones (see edge devices)
- model deployment, 248-251
- model explainability, 145, 154-158
- model lifecycle, 262
- Model Zoo, 297-299
- model.joblib, 210
- Molino, Piero, 360-362
- monitoring, 6, 161-185
 - (see also logging)
 - basics, 161
 - basics of model monitoring, 174-175
 - as DevOps best practice, 6
 - metrics for, 174
 - monitoring drift with AWS SageMaker, 175-182
 - monitoring drift with Azure ML, 182-184
 - observability and, 172-184
 - observability for cloud MLOps, 163
- Moore's Law, defined, 372

N

- natural language processing (NLP), 190
- network

- physical home network, 394
- power management for home networking, 395
- for working remotely, 394
- NLP (natural language processing), 190
- normal distributions, 37-41

O

- O'Reilly, Tim, 367
- observability
 - Application Insights, 253
 - monitoring and, 172-184
- ONNX (Open Neural Network Exchange), 296-310
 - converting Core ML models into, 310-314
 - converting PyTorch into, 299-301
 - converting TensorFlow into, 303-307
 - creating a generic ONNX checker, 301-303
 - deploying to Azure, 307-310
 - edge integration with ORT, 314-315
 - Model Zoo, 297-299
 - packaging for ML models, 95-99
- open source AutoML solutions, 151-154
 - FLAML, 153-154
 - Ludwig, 151
- operationalization, defined, 372
- optimization, 41-49

P

- packaging, 95-99
- Pandas, 39
- pear (PPEAR) revenue strategy, 399-402
 - autonomy, 401
 - exponential potential of projects, 401
 - passive income, 400
 - rule of 25%, 401
 - work as positive experience, 400
- pip, defined, 372
- pipelines, 259
 - (see also cloud pipelines)
 - Azure ML pipelines, 257-262
 - building an MLOps pipeline from zero, 56-63
 - publishing, 259
- platform automation, 16
- ports, defined, 372
- PPEAR (see pear revenue strategy)
- Professional Cloud Architect (GCP certification), 391

- Professional Machine Learning Engineer (GCP certification), 391
- project management (see technical project management)
- project plan (in technical project management), 427
- Prometheus, defined, 372
- pylint, defined, 372
- PyPI, defined, 372
- pytest, defined, 372
- Python
 - Azure CLI and Python SDK, 236-238
 - command line tools, 321-331
 - crash course, 33-35
 - energy inefficiency of, 34
 - logging different applications, 170-172
 - logging in, 165-172
 - machine learning project structure, 8-13
 - minimalistic tutorial, 36
 - MLOps Cookbook, 209-222
 - modifying log levels, 169
 - project scaffold, 8
 - requirements file, 320
 - slowness of, 33
 - testing/linting code, 32
- Python functions, 36, 194
- Python SDK, 236-238
- Python virtual environment, defined, 373
- PyTorch, converting into ONNX, 299-301

R

- Red Hat, 69
- regional job markets, 424
- registration, Azure, 243-244
- registries, 68
- reinforcement learning, 52
- "remote first" education, 423
- remote work, 393-397
 - equipment for, 394-396
 - health issues, 395
 - home work area, 395
 - home workspace virtual studio setup, 396
 - location, 396
 - network, 394
- requirements.txt, 210, 320
- requirements.txt file, 10, 320
- Ridley, Matt, 125, 367
- rule of 25% (income), 401
- rule of 25% (MLOps), 20

- run commands, 26
- runtime configuration, 259

S

- S3 (see AWS S3)
- SageMaker, 16
 - model monitoring, 163
 - monetizing MLOps, 90
 - monitoring drift with, 175-182
 - pipeline creation, 108-110
- SageMaker AutoPilot, 146-150
- SAM (AWS Serverless Application Model)
 - AWS Lambda recipes, 223-229
 - AWS Lambda-SAM containerized deploy, 224-229
- SAM Local, 223
- science, engineering versus, 2
- script, writing a, 28
- self-handicapping, 118
- serverless (term), 373
- Serverless Application Model (see SAM)
- serverless methodology
 - AWS, 194-198
 - creating a serverless function, 333-338
 - as recommended MLOps design pattern, 366
- service principal, 238-240
- SHAP, 155, 156
- shell
 - Bash shell and commands, 26-29
 - cloud shell development environments, 24-26
 - defined, 26
 - shell script, writing a, 28
- Silver, Nate, 359
- Simon, Julien, 231-233
- Sinclair, Upton, 118
- Spark-centric design, 366
- SQL-related certifications, 391
- Sqor sports social network case study, 350-355
 - Athlete Intelligence (AI product), 353-355
 - influencer rank, 352
 - Mechanical Turk data labeling, 351
- SQS queue, defined, 373
- SSH access, 72
- statistics, descriptive, 37-41
- supervised machine learning, 50
- swagger, defined, 373

T

- Taleb, Nassim, [359](#)
- target dataset, baseline dataset versus, [179](#)
- task tracking (in technical project management), [429](#)
- technical communication (DevOps best practice), [6](#)
- technical portfolio, building a, [403-408](#)
 - project example: cloud native ML application or API, [406](#)
 - project example: Docker and Kubernetes container project, [404](#)
 - project example: edge ML solution, [405](#)
 - project example: serverless AI data engineering pipeline, [405](#)
 - strategies for getting a job, [407](#)
- technical project management, [427-429](#)
 - as DevOps best practice, [6](#)
 - project plan, [427](#)
 - task tracking, [429](#)
 - weekly demo, [428](#)
- technology certifications (see certifications)
- TensorFlow
 - converting into ONNX, [303-307](#)
 - TFHub, [85](#)
- TensorFlow Developer Certificate, [391](#)
- TensorFlow Playground, [49](#)
- TensorFlow Processing Unit (see TPU)
- "10X better" education system, [421-425](#)
- Terrell, Dave, [360](#)
- testing
 - automated checks, [113](#)
 - continuous improvement and, [114](#)
 - linting, [114](#)
 - model deployment, [112-115](#)

- Python code, [32](#)

- TFHub (TensorFlow Hub), [85](#)
- theory of competitive advantage, [190](#)
- Thiel, Peter, [421](#)
- token-based authentication, [240](#)
- TPU (TensorFlow Processing Unit)
 - Coral Project and, [81-84](#)
 - porting over non-TPU models, [86-89](#)
- troubleshooting
 - Application Insights, [253](#)
 - debugging locally, [254-257](#)
 - deployment issues, [251-257](#)
 - retrieving logs, [252](#)

U

- unsupervised machine learning, [50-52](#)
- USB Accelerator, [81](#)
- utilscli.py, [210](#)

V

- versioning, of datasets, [245](#)
- Vertex AI, [268](#)
- virtual environment, Python, [11](#)
- virtual machines
 - containers versus, [68](#)
 - defined, [373](#)
- vision (see computer vision)

Y

- YAML, defined, [373](#)

Z

- Zhang, Feng, [121](#)
- ZSH, [26, 27](#)