

**Framework and Patterns for Machine Learning as Microservices using Open
Source Tools and Open Data**

by

Istvan Barabasi

Submitted in partial fulfillment
of the requirements for the degree of
Doctor of Professional Studies
in Computing Studies

at the

Seidenberg School of Computer Science and Information Systems
Pace University

Aug 2020

ProQuest Number:28091455

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent on the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



ProQuest 28091455

Published by ProQuest LLC (2020). Copyright of the Dissertation is held by the Author.

All Rights Reserved.

This work is protected against unauthorized copying under Title 17, United States Code
Microform Edition © ProQuest LLC.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

We hereby certify that this dissertation, submitted by Istvan Barabasi, satisfies the dissertation requirements for the degree of *Doctor of Professional Studies in Computing* and has been approved.

Ronald Frank

Ronald Frank (Aug 15, 2020 13:56 EDT)

Dr. Ronald Frank
Chairperson of Dissertation Committee

Date

Charles Tappert

Charles Tappert (Aug 15, 2020 13:58 EDT)

Dr. Charles Tappert
Dissertation Committee Member

Date

Lixin Tao

Dr. Lixin Tao
Dissertation Committee Member

Date

Seidenberg School of Computer Science and Information Systems
Pace University

Abstract

Framework and Patterns for Machine Learning as Microservices using Open Source Tools and Open Data

by
Istvan Barabasi

Submitted in partial fulfillment
of the requirements for the degree of
Doctor of Professional Studies
in Computing

Aug 2020

Machine Learning is part of our everyday life, including social media analytics, online shopping, stocks performance prediction and other areas.

Machine Learning is a complex process by itself. Explained at high level, takes data from around us and applies algorithms to it in order to generate intelligent outcomes using machine-based computation.

There are multiple challenges with using and practicing Machine Learning, such as new technology adoption and handling constraints around accessing and using data, running computational workloads, using multiple cloud platforms, providers and proprietary technologies.

This research explores new concepts and practical approach implementing and using machine learning in an easy, scalable, and sustainable way to solve common everyday problems. The research proposes adopting a machine learning framework defined using microservices architecture style, together with a series of patterns that provide practical guidance and are applicable for a broad set of machine learning tools and algorithms.

Examples within this research refer to real estate market, specifically using open data available regarding single family real estate properties and real estate loans performance. Use cases and sample practices include real estate property data acquisition and real estate properties price prediction using machine learning patterns.

Acknowledgements

First, I would like to thank my academic advisor - Dr. Ronald Frank for all the help, patience, support and guidance provided over the years at the Seidenberg School.

I would also like to thank dissertation committee members Dr. Charles Tappert and Dr. Lixin Tao for their support and advise provided throughout the DPS education years.

In addition, I am deeply grateful to my family for their patience and support over all these years, while attending the Place University DPS program and working on this dissertation research.

Finally, I would like to thank all DPS faculty and staff who have advised and helped me a great deal on this journey.

Table of Contents

Abstract	iii
Acknowledgements	iv
List of Figures	ix
Chapter 1 Introduction	1
1.1 Background	1
1.2 Problem Statement	2
1.3 Solution Strategy	4
1.4 Research Contributions	4
1.5 Solution Methodology	5
1.6 Dissertation Roadmap/Outline	6
1.7 Conclusion	7
Chapter 2 Technology, Frameworks and Methods Review	9
2.1 Landscape of current Machine Learning Frameworks and Platforms	9
2.2 Technology Landscape	10
2.3 Machine Learning with Nvidia GPU Platform	13
2.4 Microservices, an Architecture Style	15
2.5 Design Patterns and Framework	16
Chapter 3 Literature Review	20
3.1 Design Patterns	20
3.2 Accelerating Computation using Graphical Processing Units	21
3.3 Using containerized workloads for machine learning	23
3.4 Using DevOps practices and patterns for machine learning	25
3.5 Practicing MLOps in a structured, sustainable and scalable way	28
3.6 Serverless Computing Framework for MLOps Workflows	29
3.7 Key industry expectations from Machine Learning Frameworks in 2019	30

3.8	ML Platform Runtimes and Frameworks efficiency estimation.....	31
3.9	Serverless technology adoption for ML is emerging cloud-native trend.....	32
3.10	Real-time deep learning frameworks as microservices are emerging	33
3.11	Serving deep learning models in serverless cloud native platform.....	34
3.12	Serverless application platform and middleware requirements	35
3.13	Machine learning as microservices with Big Data	36
3.14	Evaluation of open source serverless computing frameworks.....	37
3.15	Stateful migration for machine learning microservices	38
3.16	High-available microservice type applications on Kubernetes.....	39
3.17	Patterns for Distributed Transactions within a microservices architecture	40
3.18	Microservices based applications hosting on Kubernetes Platform	41
3.19	Scaling machine learning workloads using distributed microservices	42
Chapter 4	Framework and Patterns for Machine Learning with Microservices.....	43
4.1	Machine Learning Overview	43
4.2	Pattern Recognition Overview in Context of Machine Learning	44
4.3	The Workflow of a Machine Learning Project.....	45
4.4	Brief introduction to Microservices	46
4.5	Virtualized Infrastructure Compute Elements for Microservices Deployment	50
4.6	Container type virtualization.....	51
4.7	Function as a Service type virtualization	51
4.8	Serverless Compute and Workloads	52
4.9	Containers vs. Virtual Machines (VMs)	53
4.10	Containers implementing Microservices	54
4.11	Serverless Functions implementing Microservices	54
4.12	Machine Learning Algorithms Library as Serverless Microservices	55
4.13	Framework for Machine Learning as Microservices	56
4.13.1	Microservices based architecture for Data Collection and Management .	58

4.13.2	Microservices based architecture for Model Training	60
4.13.3	Microservices based architecture for Model Serving Components	62
4.14	Proposed Machine Learning Framework and Patterns as Microservices	63
Chapter 5	Machine Learning Catalog as Live Microservices	67
5.1	Overview of the Machine Learning Catalog.....	67
5.2	MLCM Catalog Implementation - High level Overview.....	68
5.3	Proposed MLCM Catalog Implementation - High level Overview.....	69
5.3.1	Sample MLCM Supervised Learning Library	70
5.3.2	Sample MLCM Unsupervised Learning Library	72
Chapter 6	Proposed Machine Learning Patterns	73
6.1	Data Acquisition	73
6.1.1	Overview.....	73
6.1.2	High-Level overview of data types and data collection activities	74
6.1.3	High-Level overview of the Data Acquisition Data Sources.....	75
6.1.4	Data Acquisition as Microservices, published as a Catalog of Services ..	79
6.1.5	Data Acquisition Pattern.....	81
6.1.6	Data Mining Pattern.....	84
6.1.7	Data Preparation and Splitting Pattern.....	88
6.1.8	Jupyter Notebooks Provisioning Pattern.....	91
6.1.9	Microservices Packaging (Docker Container or Function)	94
6.1.10	Cloud Native Compute Platforms Provisioning for Microservices	98
6.1.11	Agile Practices for MLM Pattern.....	102
6.1.12	Model Training Pattern	105
6.1.13	Model Testing Pattern.....	109
6.1.14	Model Serving Pattern: Definition.....	111
6.1.15	Features Driven Prediction Pattern	113
6.1.16	Models Conversion Pattern: Definition	116

6.1.17	Artifacts Persistence Pattern: Definition.....	120
6.1.18	Cloud Native Machine Learning Catalog: Definition.....	124
6.1.19	Compute Acceleration for Microservices using GPUs.....	129
6.1.20	Micro Allocation of HW resources to Microservices: Definition	135
6.1.21	Cloud Native Data Catalog for Machine Learning: Definition	141
6.1.22	Multi-Cloud & Site Machine Learning Workloads Migration	146
6.1.23	Maintaining the identity and state of Machine Learning Workloads	154
6.2	Applied Patterns in the area of Real Estate Properties and Price Prediction ..	157
6.2.1	Applied Data Acquisition Pattern	157
6.2.2	Applied Pattern for Simplified Data Mining and ML Model Training ..	163
6.2.3	Applied Pattern for Extended Data Mining and ML Model Training	169
Chapter 7	Future Work.....	177
	Glossary of Terms and Acronyms	179
	Glossary of Referenced Technology Links.....	198
	References.....	202

List of Figures

Figure 1: RAPIDS, GPU-enabled open data science platform (image from [132])	15
Figure 2: Containerized DevOps architecture for Machine Learning	24
Figure 3: Containers based multi-cluster, multi-ML model (image from ref [16])	24
Figure 4: Data Science Lifecycle activities and iterative execution	27
Figure 5: MLOps infrastructure and activity requirements (image from ref [3])	29
Figure 6: Serverless ML workloads outperforms traditional (image from ref [5])	30
Figure 7: Top ML Framework usability features for the industry (image from ref [6]) ...	31
Figure 8: Deep learning framework as microservices at edge (image ref [12])	33
Figure 9: Serverless apps requirements, degree of abstraction (image from ref [15])	36
Figure 10: Analytics and Big Data Platform as microservices (image from ref [19])	37
Figure 11: Comparison for open source serverless frameworks (image from ref [20]) ...	38
Figure 12: High-Level workflow of a machine learning project	46
Figure 13: Conceptual microservices design template for an ML project	47
Figure 14: Microservices implementation using Cloud Native Platforms	48
Figure 15: Scalable Microservices deployment using Cloud Native Platforms	49
Figure 16: Business application deployment model using microservices	50
Figure 17: Two types of virtualization: Virtual Machines vs Containers	53
Figure 18: High-level overview for generic machine learning framework components ..	57
Figure 19: Data collection and management using microservices	59
Figure 20: Practical example: Data collection for Zillow.com listed properties	59
Figure 21: ML Model Training using containerized microservices	60
Figure 22: ML Model Training with microservices as serverless functions	61
Figure 23: ML Model Serving with microservices as containers deployment	62
Figure 24: ML Model Serving with microservices as cloud functions deployment	63

Figure 25: Overview of the proposed ML Framework as Microservices	66
Figure 26: Fishbone diagram for Data Acquisition Data Sources	76
Figure 27: Generic Structure for a Data Collector.....	82
Figure 28: Generic Structure for a Data Mining Microservice using Jupyter Notebook..	85
Figure 29: Data Mining Microservice based on clasification use cases	87
Figure 30: Data Mining Microservice based on clustering use cases.....	88
Figure 31: Generic structure for a Data Mining Microservice using Jupyter Notebook ..	89
Figure 32: Data Preparation use cases	90
Figure 33: Data Splitting use cases.....	91
Figure 34: Generic Structure for the Jupyter Notebooks Provisioning as a Container.....	92
Figure 35: Generic Structure for the Jupyter Notebooks Provisioning as a Container.....	93
Figure 36: Microservice with Docker image deployent as Cloud Native Container	96
Figure 37: Containerized microservice provisioning as Cloud Native Function.....	96
Figure 38: Generic program, CLI or script deployment as a Cloud Native Function	96
Figure 39: Cloud Native Compute Platform for a four node Kubernetes cluster	99
Figure 40: Cloud Native Compute Platform for a single node Mini-Kubernetes cluster .	99
Figure 41: GitOps based CI/CD practices applied to the MLM Framework.....	104
Figure 42: Model Training as a Service deployed as serverless function.....	107
Figure 43: Model Testing as a Service using serverless function.....	110
Figure 44: Structure for Model Serving as Microservice deployed as Cloud Function .	112
Figure 45: Generic Structure for the interactice Features Driven Prediction Service	115
Figure 46: Generic Structure for the interactice Features Driven Prediction Service	118
Figure 47: Portfolio of MLM Artifact Repositories and Registries.....	121
Figure 48: Machine Learning as Live Microservices (MLCM) Catalog of Functions...	126
Figure 49: High-level overview of MLM compute acceleration using GPUs.....	132
Figure 50: Micro-Allocation of hardware resources for ML as microservices	137
Figure 51: Apache Hive as microservice for Cloud Native Data Catalog.....	142

Figure 52: Hive as single catalog and data access for Machine Learning (CNDC)	145
Figure 53: Distributed CockroachDB datastore for ML, managed using Hive	146
Figure 54: Data Science Project stages performed in select locations or clouds.....	148
Figure 55: Core MLM Framework services distributed across multi-clouds and sites ..	149
Figure 56: Workloads management & migration via Jenkins-x or Jupyter	150
Figure 57: Managing secrets and credentials in support of ML Workflows migration..	152
Figure 58: Standard set of artifact repositories and registries required for each POP	153
Figure 59: MinIO storage deployed to GKE – source of picture from GCP	156
Figure 60: Generic structure for a Data Collector deloyed using microservices	158
Figure 61: Enterprise Relationship Diagram (ERD) for real-estate data sources	160
Figure 62: Data structure for pattern MOD-TRAIN-003	166
Figure 63: Top 15 features to predict property prices (image from ref [13])	170
Figure 64: Extended data structure for pattern MOD-TRAIN-004	174

Chapter 1

Introduction

1.1 Background

Machine Learning (ML) is an application of artificial intelligence (AI) that automatically learns from data and is able to make guesses and predictions that are statistically verifiable as accuracy and subsequently prove good enough to be applicable and useful. ML represents data analytics practices and patterns that teach computers to do what humans and animal perform naturally, such as learn from prior experience, use evidence to inference and abstract. ML models are established by learning directly from the data and in some cases taking input from humans or other machines (ex: supervised learning).

Machine Learning uses algorithms to learn from data, perform statistical processing using the data and performs subsequent predictions. Modern implementations of ML develop and harvest ML Models and produce programs that can be implemented and executed as jobs. Jobs will consume data and are implemented on various platforms, such as in-house (“on-premise”) computers and/or on public cloud platforms.

The proposed ML Framework and Patterns are addressing key practical elements of applied ML, with focus on simplification, streamlining and making it accessible for everyday practitioner to easier implement and produce practical results.

Pattern Recognition (PR) is the process of recognizing patterns using Machine Learning Algorithms. PR helps detect characteristics of data, including features, classes, clusters and other properties that yield information about a given system, service or studied entity.

Microservices represent engineering patterns for developing applications as a suite of small, independently deployable services built in alignment with key business functions of a solution. Microservices enable simplification and streamlining, while open-source and cloud-native technologies will provide accessibility and ability to implement on-premise and/or public-cloud agnostic. The proposed ML framework includes practices for developing, training and deploying deep learning models as microservices.

1.2 Problem Statement

The multitude of today's machine learning frameworks, together with the broad spectrum of computing platforms and associated technologies for virtualization, computation acceleration and real-time streaming make ML projects implementation very complex and hard to accomplish at scale and as a service.

This is made worst by presence of multiple public cloud services providers that use different and frequently proprietary technologies that can easily lock-in customers and make them cloud and platform dependent.

Every major public cloud provider offers proprietary services for Machine Learning:

- Amazon Web Services (AWS) offers ML services for SageMaker Framework,
- Microsoft Azure Cloud provides ML services based on MXNet Framework,

- IBM Cloud offers Watson Studio and Watson Machine Learning services,
- Google provides an AI Hub platform, implemented using its Cloud ML Engine.

Often proprietary technology will lock-in developers and data scientists to sub-optimal and expensive solutions, thus negatively impacting their research, machine learning practices and implementation efforts.

The problem my dissertation addresses is the ability to practice machine learning in structured, simple, repeatable, affordable and “compute anywhere” platform independent way. Researchers and machine learning practitioners need easy to use, readily available and cloud platform independent technology and machine learning practices to work with data and solve everyday problems that impact our lives. Everyday ML problems include predicting price of a real estate, predicting performance of stocks, analyze data collected for an area of interest and others.

My dissertation addresses the above listed ability factors and problem criteria as follows:

- Structured: define a Machine Learning Framework,
- Simple: instantiate framework components as microservices,
- Repeatable: define patterns, as part of the proposed framework,
- Affordable: use light-weight containers or transient “serverless” functions,
- Compute anywhere: use cloud native computing platforms and virtualization,
- Platform independent: compute engine, operating system and cloud agnostic,
- Readily available technology: use of open source tools,
- Readily available data: use of public and community provided open data sources.

1.3 Solution Strategy

The solution strategy entails researching and developing a framework and underpinning patterns that leverage the following elements:

- REST-based microservices architecture and software development patterns,
- Light-weight virtualization components, that implement simplified software stack for various ML frameworks, such as Nvidia CUDA Stack, RapidsAI Open GPU Data Science and others,
- Cloud agnostic big data components that enable seamless parallel and distributed computing functionality and scalability,
- Open source technologies including Big Data, virtualization and ML frameworks,
- Cloud Native tools and technologies that enable cloud provider agnostic solutions,
- Leverage open source tools that enable DevOps and MLOps practices and focus on applying them for developing and testing Machine Learning patterns and practices. This includes continuous integration and continuous development (CI/CD) tools and workflow automation tools to enable 360-degree iterative coverage for data science and machine learning projects lifecycle.

1.4 Research Contributions

My research contributions within this dissertation document include the following:

- Machine learning framework, that leverages REST API based microservices and uses

- Lightweight virtual compute platforms: containers and serverless functions,
 - CPUs and GPU compute resources, both as physical and virtual.
- Live Catalog of Machine Learning Microservices as cloud native functions.
- Cloud Native Data Catalog (Metadata as a Service for Machine Learning).
- ML workloads migration across Cloud Native Platforms: public clouds and on-site.
- Microservices patterns and practices for:
 - Data acquisition, preparation, processing and transfer using pipelines,
 - Examples using dataset for real estate and mortgage loans,
 - Neural networks architecture, model training, model lifecycle management,
 - Deploying and using Cloud Native Compute Platforms.
- Patterns for compute resource micro allocation to individual microservices.
- Applied ML patterns and practices for real-estate data analysis and prediction.
- Patterns for compute resource micro allocation to individual microservices, with the ability to flexible scale their allocation as microservices and workloads demand.

1.5 Solution Methodology

The methodology applied within the dissertation includes the following key elements:

- Identify, categorize and prioritize the main challenges to achieve the main objective,
- Define what the referenced patterns, practices, technology elements are as definition and provide representative examples,
- Identify key characteristics of quality data acquisition, preparation, processing, patterns-based analysis and prediction,
- Define framework and practices for data acquisition, machine learning and evaluation of results,
- Describe in depth technology elements and open source software used for data acquisition and computation,
- Select and use open technology and software readily available to average IT, data analyst and entry level data science personnel,
- Link patterns and artifacts together into cohesive framework and workflows
- Provide feasibility validation for key patterns and framework elements.

1.6 Dissertation Roadmap/Outline

The Dissertation covers two key areas which jointly provide solution to the problem statement presented earlier:

- Machine learning (ML) patterns in context of microservices architectural style
- Applied examples for the introduced ML patterns in the area of real estate data collection and property price prediction.

High-level outline of the chapters and overall content:

- Chapter-1: Introduction and Background
- Chapter-2: Technology, Frameworks and Methods Review
- Chapter-3: Literature Review
- Chapter-4: Framework and Patterns for Machine Learning with Microservices
- Chapter-5: Proposed Machine Learning Patterns
- Future Work
- Glossary of Terms and Acronyms
- Glossary of Referenced Technology Links
- References

1.7 Conclusion

Implementing Machine Learning as microservices (containers, functions) has clear benefit and impact in providing simple, repeatable, structured and platform independent practices.

All introduced and documented machine learning patterns and practices can be deployed and utilized as microservices, such as containers and /or cloud functions.

- Data collection and acquisition patterns should use primarily container type packaging as deployment option,
- Machine learning patterns, such as Model Training and Model Serving, should primarily be deployed as cloud functions.

The defined patterns should be implemented on Cloud Native Compute Platforms:

- Data Scientist's Laptops, using Kubernetes Minikube edition or Docker Desktop Application with its embedded Kubernetes runtime,
- Developer Teams should use small K3S environment with one master and 2-3 worker nodes. K3S nodes deployment is automated using Rancher Inc's K3D.

Machine Learning workloads can be implemented and deployed as cloud and platform agnostic and can be migrated across cloud providers.

- Requirements to achieve include:
 - Several database /storage repositories
 - Record of reference datasets or service registry

Machine Learning artifacts can be organized and instantiated as:

- Live machine learning catalog, deployed as serverless functions,
- Machine learning Data Catalog with data acquisition pipelines that can be deployed as microservices.

Chapter 2

Technology, Frameworks and Methods Review

2.1 Landscape of current Machine Learning Frameworks and Platforms

The landscape of machine learning frameworks has substantially changed within the last couple of years. A few years ago, none of the current popular framework were around, except Scikit-Learn and Theano.

Nowadays there is a large variety of ML technologies and frameworks; a few of them are mentioned below:

- **Tensorflow**, developed by Google, the most popular ML framework these days
- **PyTorch** was developed for Facebook, adopted by Twitter, Salesforce and others
- **Sonnet** is built on top of Tensorflow, developed by DeepMind Company, who later was acquired by Google. Sonnet is used to create neural networks for many different purposes (un/supervised learning, reinforcement learning, and others).
- **Keras** is an ML Framework that works as a high-level API on top of other popular networks, such s Theano, CNTK, Tensorflow.
- **MXNet** is a highly scalable deep learning framework, that supports a large number of languages. Supports multiple GPUs with optimized computation and rapid context switching.
- **ONNX** is a project resulting from Microsoft and Facebook's collaboration with attempt to establish an open format to exchange trained machine learning models.
- **Apache Spark/MLlib** is a collection of ML algorithms and features extraction

- **Deep Learning 4J (DL4J)**, runs on a Apache Spark middleware and leverages distributed computing with Apache Spark and Hadoop to accelerate training.
- **Rapids.ai** CUMML library implements ML algorithms and mathematical primitive functions for the RAPIDS Open GPU Data Science project [132]
- **XGBoost** is a ML Library designed for training decision trees and random forests.

2.2 Technology Landscape

Virtualization is one of the key technologies for cloud computing. Virtualization in the cloud covers multiple areas:

- **Compute**: includes virtual machines, containers and functions that use CPU(s),
- **Compute acceleration using GPUs**: subset of cloud providers introduced GPU, virtualization into their compute services (ex: VM, containers with virtual GPU),
- **Networking**: necessary for all components to communicate. Networking has multiple virtualization implementations, such as VLANs, Overlay Networks, VPCs (virtual private clouds) or VDCs (virtual data centers),
- **Storage**: covers virtualized storage attached to virtual machines / compute and unattached storage exposed as API etc.
- **Data abstraction** as virtual data warehouse and use of metadata to consume data, find datasets and use schemas. Example of such abstraction is using Hive with Hadoop.

Cloud Computing is type of shared multi-tenant computing which usually provides a remote pool of computing resources and compute services via the internet. The representative cloud compute services are the following:

- Infrastructure as a Service (IaaS), which provides virtualized compute (virtual machines with dedicated operating system), virtualized storage and virtualized network (virtual LANs; VLANs).
- Container as a service (CaaS) also provides virtualized compute which is abstracted away from operating systems using container packaged runtime environments which only include libraries and applications. Containers don't have dedicated operating system. Containers communicate with the operating systems kernel, as such they inherit a series of process management and security functions.
- Function as a Service (FaaS) , which is a lightweight, event-based, asynchronous compute solution that allows creating and running small single-purpose functions that can be instantiated within the cloud without the need to instantiate a server or a runtime environment.
- Jobs, also called "cloud runs", are stateless containers instantiated using HTTPS requests. Jobs can be deployed using container services (ex: using Google Kubernetes Engine) or as serverless functions (ex: using Google Cloud Run).
- Services handling computer jobs sometimes are called Jobs as a Service (JaaS).

Within my dissertation, as choice of cloud compute environment, in addition to on-premise compute, I will focus on Kubernetes container services, which is the latest evolution of CaaS. Public cloud providers offering this service are: [117]

- Amazon Web Services (AWS): Kubernetes on AWS
- Microsoft: Azure Kubernetes Service (AKS)
- Google: Google Kubernetes Engine (GKE)

The “**Cloud-Native**” concept emerged within the last 4 years and is characteristic to service or applications that can leverage properties and interfaces offered by cloud computing providers or frameworks, but their code packaging libraries are independent from the cloud provider platforms.

Cloud native applications leverage elements from the “The Twelve-Factor App” [143] definitions which promotes concepts for optimal design and deployment in the cloud environment such as declarative formats to setup automation, clean contract with operating system and/or cloud provider, continuous build and deployment and minimum difference between development and production environments.

While concepts of cloud computing implementations are similar, cloud service providers use many proprietary technologies and solution in their cloud environments. These proprietary technologies include Big Data, Machine Learning, Deep Learning, DevOps and other cloud services areas. This makes difficult migrating machine learning applications between cloud providers, as well as between on-premise solutions and cloud providers.