**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

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 — Date
Chairperson of Dissertation Committee

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

Dr. Charles Tappert — Date
Dissertation Committee Member

Lixin Tao

Dr. Lixin Tao — Date
Dissertation Committee Member

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 the 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

# List of Figures

# 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 of 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 of a Apache Spark middleware and leverages distributed computing with Apache Spark and Hadoop to accelerate training.
- **Rapids.ai** CUML 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.