PERFORMANCE AND SCALABILITY WITH OPTIMIZED MACHINE LEARNING LIBRARIES

DR. FABIO BARUFFA

SENIOR TECHNICAL CONSULTING ENGINEER, INTEL IAGS

Legal Disclaimer & Optimization Notice

Performance results are based on testing as of September 2018 and may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.

INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.
Notice revision #20110804

LEGAL NOTICES & DISCLAIMERS

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer. No computer system can be absolutely secure.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit http://www.intel.com/performance.

Cost reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.

Statements in this document that refer to Intel's plans and expectations for the quarter, the year, and the future, are forward-looking statements that involve a number of risks and uncertainties. A detailed discussion of the factors that could affect Intel's results and plans is included in Intel's SEC filings, including the annual report on Form 10-K.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.

Intel, the Intel logo, Pentium, Celeron, Atom, Core, Xeon, Movidius and others are trademarks of Intel Corporation in the U.S. and/or other countries.
*Other names and brands may be claimed as the property of others.

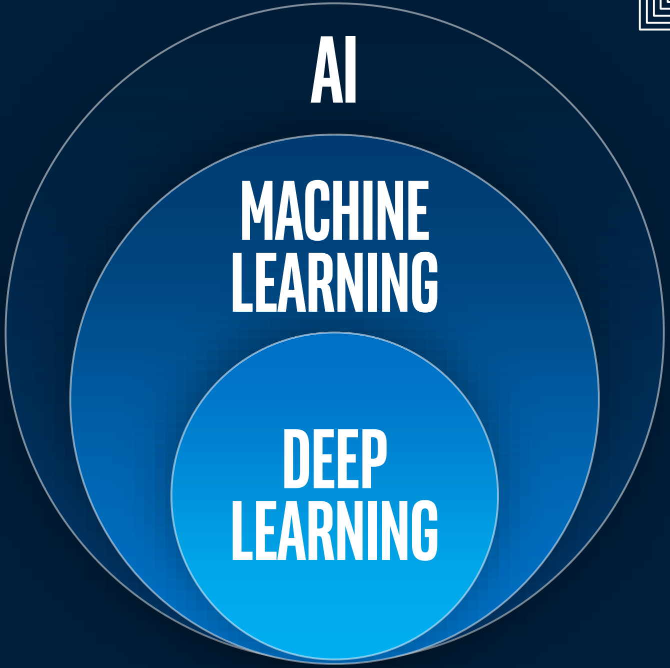
Outline

- Introduction to the AI problem
- Intel® AI optimized software stack
- Integration with the popular AI/ML frameworks:
  - Scikit-learn accelerate with Intel® Data Analytics Acceleration Library (Intel® DAAL) and DAAL4Py
- Hands-on

INTRODUCTION

WHAT IS AI?

- Regression
- Classification
- Clustering
- Decision Trees
- Data Generation
- Image Processing
- Speech Processing
- Natural Language Processing
- Recommender Systems
- Adversarial Networks



SUPERVISED LEARNING

UNSUPERVISED LEARNING

REINFORCEMENT LEARNING

NO ONE SIZE FITS ALL APPROACH TO AI

HOW DOES MACHINE LEARNING WORK?

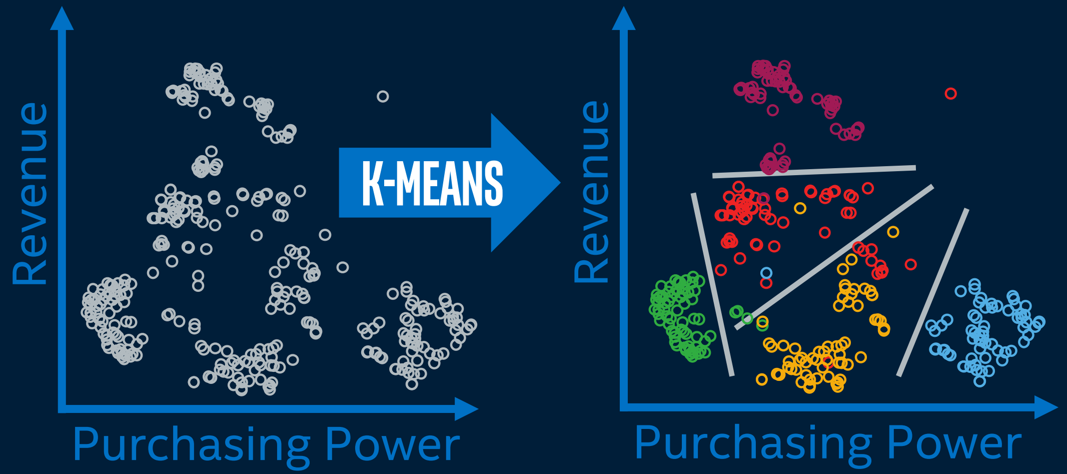
MACHINE LEARNING

- Regression
- Classification
- **Clustering**
- Decision Trees
- Data Generation

DEEP LEARNING

- Speech Processing
- Natural Language Processing
- Recommender Systems
- Adversarial Networks
- Reinforcement Learning

MACHINE LEARNING (CLUSTERING)



CHOOSE THE BEST AI APPROACH FOR YOUR CHALLENGE

HOW DOES DEEP LEARNING WORK?

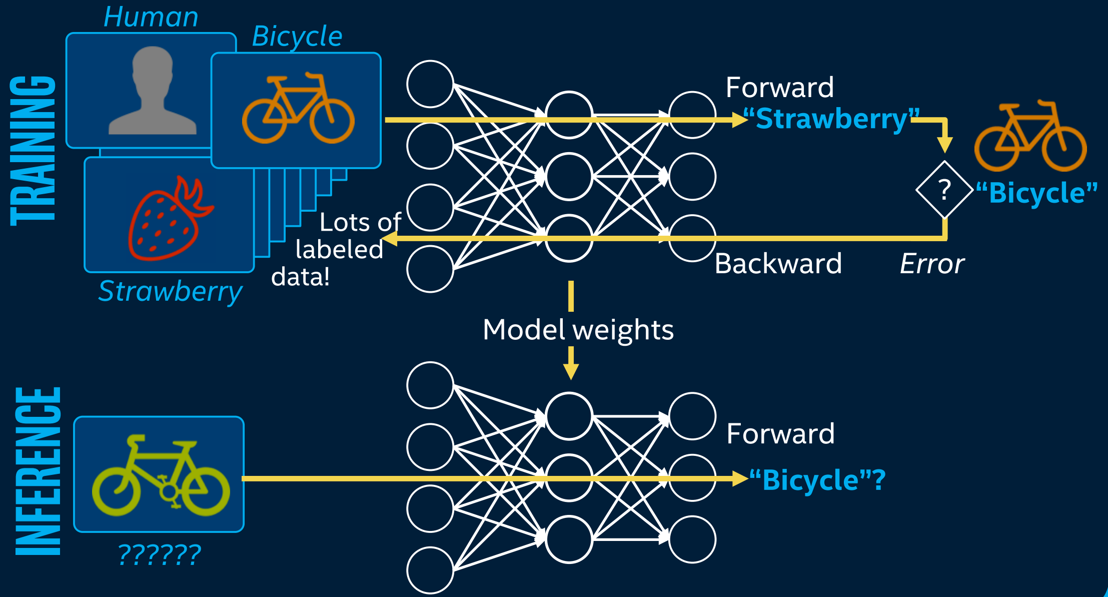
MACHINE LEARNING

- Regression
- Classification
- Clustering
- Decision Trees
- Data Generation
- Image Processing

DEEP LEARNING

- Speech Processing
- Natural Language Processing
- Recommender Systems
- Adversarial Networks
- Reinforcement Learning

DEEP LEARNING (IMAGE RECOGNITION)

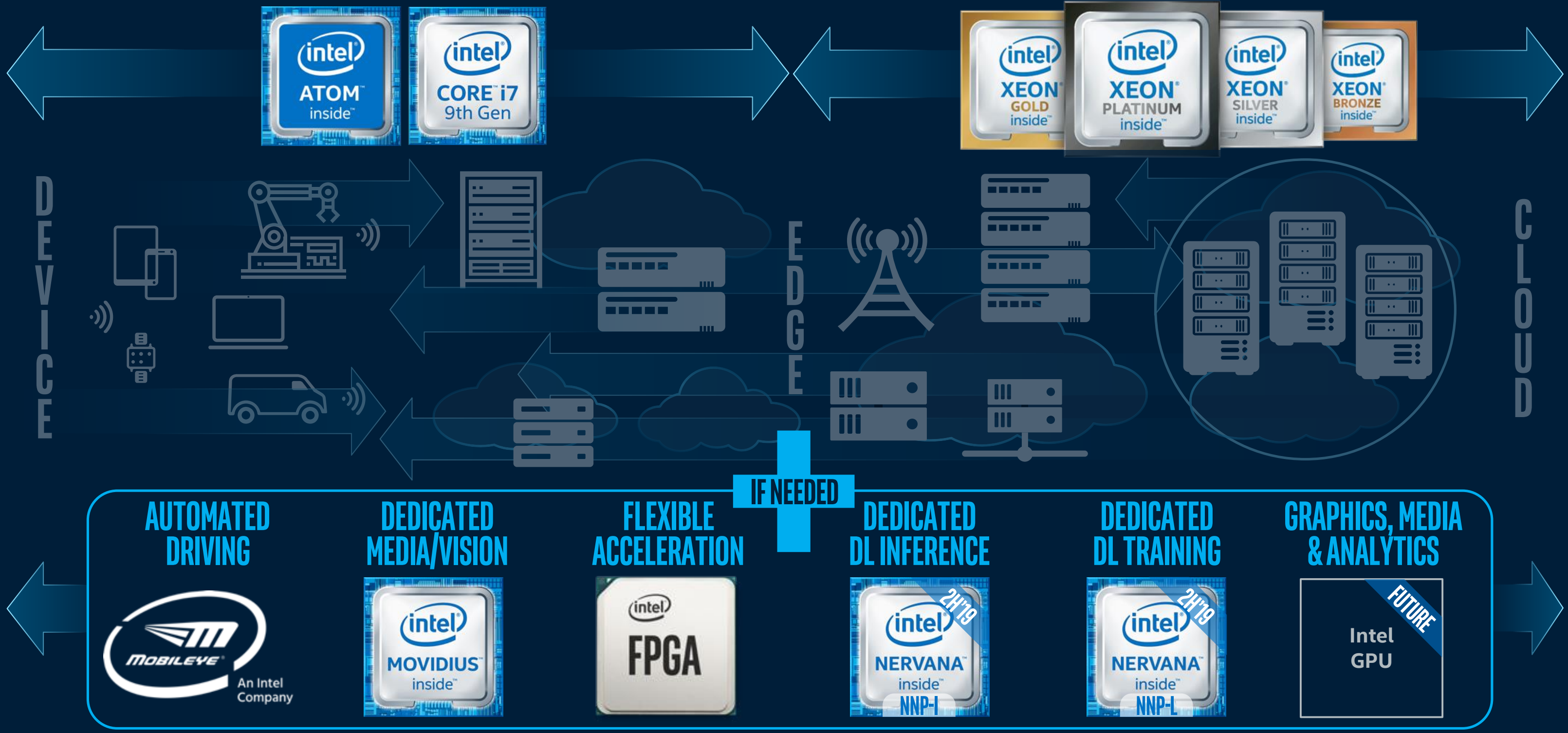


CHOOSE THE BEST AI APPROACH FOR YOUR CHALLENGE

DEPLOY AI ANYWHERE

WITH UNPRECEDENTED HARDWARE CHOICE

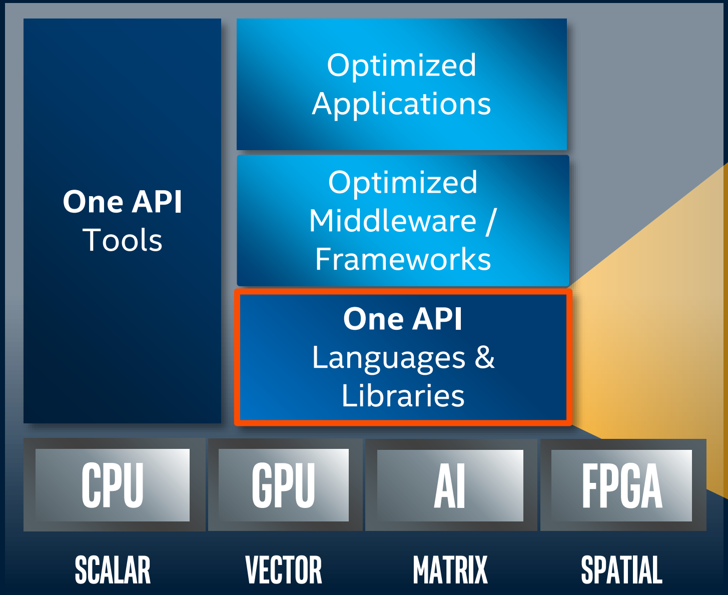
Visit: www.intel.ai/technology



All products, computer systems, dates, and figures are preliminary based on current expectations, and are subject to change without notice.

REVOLUTIONIZING PROGRAMMABILITY

INTEL'S ONE API



Optimized Applications

Optimized Middleware / Frameworks

One API Tools

One API Languages & Libraries

CPU — SCALAR

GPU — VECTOR

AI — MATRIX

FPGA — SPATIAL

DATA PARALLEL C++

Based on C++ and uses C / C++ constructs

Incorporates SYCL* for data parallelism & heterogeneous programming

Language extensions driven through an open community project

First available – Q4 2019

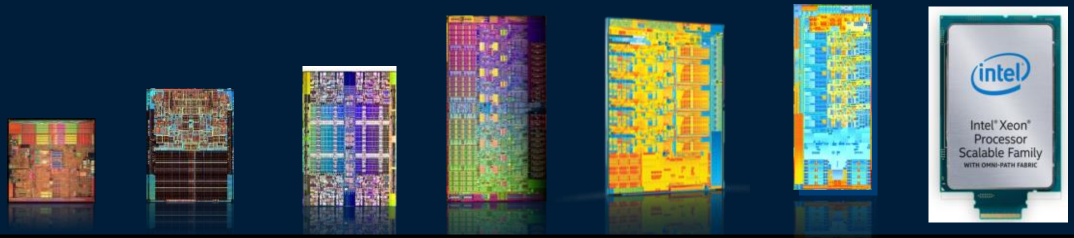
* from the Khronos Group

OPEN & INDUSTRY STANDARDS, UNCOMPROMISED PERFORMANCE, INTEROPERABLE

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

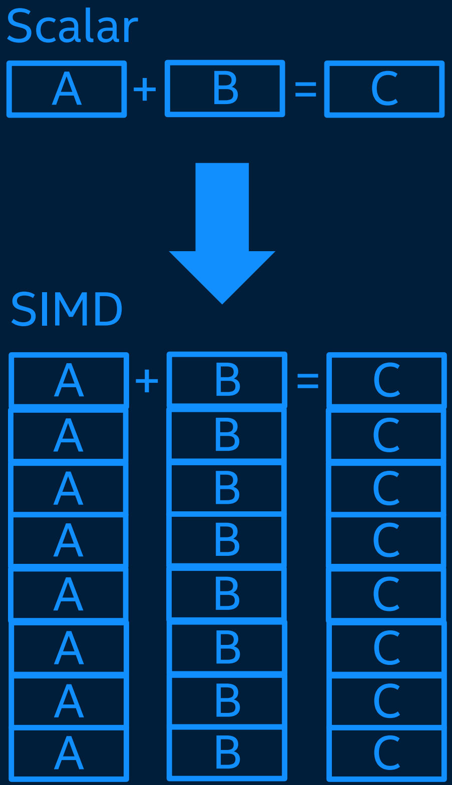
THE EVOLUTION OF MICROPROCESSOR PARALLELISM

More cores → More Threads → Wider vectors



	Intel® Xeon® Processor 64-bit	Intel® Xeon® Processor 5100 series	Intel® Xeon® Processor 5500 series	Intel® Xeon® Processor 5600 series	Intel® Xeon® Processor E5-2600 v2 series	Intel® Xeon® Processor E5-2600 v3 series v4 series	Intel® Xeon® Scalable Processor[1]
Up to Core(s)	1	2	4	6	12	18-22	28
Up to Threads	2	2	8	12	24	36-44	56
SIMD Width	128	128	128	128	256	256	512
Vector ISA	Intel® SSE3	Intel® SSE3	Intel® SSE4- 4.1	Intel® SSE 4.2	Intel® AVX	Intel® AVX2	Intel® AVX-512

[1] Product specification for launched and shipped products available on ark.intel.com.



Scalar

A + B = C

SIMD

A + B = C
A B C
A B C
A B C
A B C
A B C
A B C
A B C

SPEED UP DEVELOPMENT

USING OPEN AI SOFTWARE

Visit: www.intel.ai/technology

MACHINE LEARNING | DEEP LEARNING

TOOLKITS
App developers

ANALYTICS ZOO
Open source platform for building E2E Analytics & AI applications on Apache Spark* with distributed TensorFlow*, Keras*, BigDL

OpenVINO™
Deep learning inference deployment on CPU/GPU/FPGA/VPU for Caffe*, TensorFlow*, MXNet*, ONNX*, Kaldi*

NAUTA
Open source, scalable, and extensible distributed deep learning platform built on Kubernetes (BETA)

LIBRARIES
Data scientists

Python
- Scikit-learn
- Pandas
- NumPy

R
- Cart
- Random Forest
- e1071

Distributed
- MlLib (on Spark)
- Mahout

Intel-optimized Frameworks

TensorFlow* Caffe2* ONNX* mxnet* PyTorch* BigDL

And more framework optimizations underway including PaddlePaddle*, Chainer*, CNTK* & others

KERNELS
Library developers

Intel® Distribution for Python*
Intel distribution optimized for machine learning

Intel® Data Analytics Acceleration Library (DAAL)
High performance machine learning & data analytics library

Intel® Math Kernel Library for Deep Neural Networks (MKL-DNN)
Open source DNN functions for CPU / integrated graphics

nGraph
Open source compiler for deep learning model computations optimized for multiple devices (CPU, GPU, NNP) from multiple frameworks (TF, MXNet, ONNX)

[1] An open source version is available at: 01.org/openvinotoolkit

*Other names and brands may be claimed as the property of others.

Developer personas show above represent the primary user base for each row, but are not mutually-exclusive

All products, computer systems, dates, and figures are preliminary based on current expectations, and are subject to change without notice.

INTEL® AI OPTIMIZED SOFTWARE

PRODUCTIVITY WITH PERFORMANCE VIA INTEL® PYTHON*

Intel® Distribution for Python*

scikit learn | NumPy | SciPy | Numba | pandas $y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$ | mpi4py | SMP | • • •

Easy, out-of-the-box access to high performance Python

- Prebuilt accelerated solutions for data analytics, numerical computing, etc.
- Drop in replacement for your existing Python. No code changes required.

Learn More: software.intel.com/distribution-for-python

INSTALLING INTEL® DISTRIBUTION FOR PYTHON* 2019

Standalone Installer

Download full installer from
https://software.intel.com/en-us/intel-distribution-for-python

Anaconda.org
Anaconda.org/intel channel

```
> conda config --add channels intel
> conda install intelpython3_full
> conda install intelpython3_core
```

PyPI

```
> pip install intel-numpy
> pip install intel-scipy
> pip install mkl_fft
> pip install mkl_random
```

+ Intel library Runtime packages
+ Intel development packages

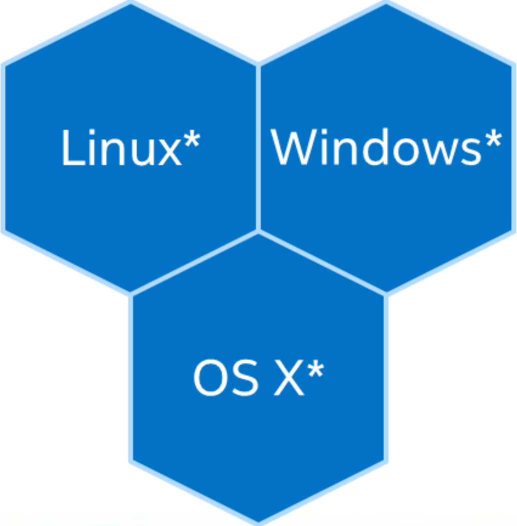
Docker Hub

```
docker pull intelpython/intelpython3_full
```

YUM/APT

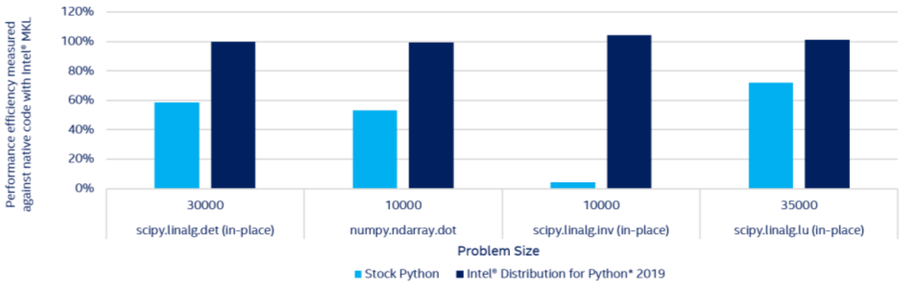
Access for yum/apt:
https://software.intel.com/en-us/articles/installing-intel-free-libs-and-python

2.7 & 3.6
(3.7 coming soon)



FASTER PYTHON* WITH INTEL® DISTRIBUTION FOR PYTHON 2019

Intel optimizations improve Python Linear Algebra efficiency closer to native code speeds on Intel® Xeon™ processors



Performance results are based on testing as of **July 9, 2018** and may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure. Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information, see Performance Benchmark Test Disclosure.

Testing by Intel as of July 9, 2018. Configuration: Stock Python: python 3.6.6 hc3d631a_0 installed from conda, numpy 1.15, numba 0.39.0, llvmlite 0.24.0, scipy 1.1.0, scikit-learn 0.19.2 installed from pip; Intel Python: Intel® Distribution for Python* 2019 Gold: python 3.6.5 intel_11, numpy 1.14.3 intel_py36_5, mkl 2019.0 intel_101, mkl_fft 1.0.2 intel_np114py36_6,mkl_random 1.0.1 intel_np114py36_6, numba 0.39.0 intel_np114py36_0, llvmlite 0.24.0 intel_py36_0, scipy 1.1.0 intel_np114py36_6, scikit-learn 0.19.1 intel_np114py36_35; OS: CentOS Linux 7.3.1611, kernel 3.10.0-514.el7.x86_64; Hardware: Intel(R) Xeon(R) Gold 6140 CPU @ 2.30GHz (2 sockets, 18 cores/socket, HT:off), 256 GB of DDR4 RAM, 16 DIMMs of 16 GB@2666MHz

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804.
For more complete information about compiler optimizations, see our Optimization Notice.

Linear Algebra functions in Intel Distribution for Python are faster than equivalent stock Python functions

High Performance Python Distribution

- Accelerated NumPy, SciPy, scikit-learn well suited for scientific computing, machine learning & data analytics
- Drop-in replacement for existing Python. No code changes required
- Highly optimized for latest Intel processors
- Take advantage of Priority Support – connect direct to Intel engineers for technical questions[2]

What's New in 2019 version

- Faster Machine learning with Scikit-learn: Support Vector Machine (SVM) and K-means prediction, accelerated with Intel® Data Analytics Acceleration Library
- Integrated into Intel® Parallel Studio XE 2019 installer. Also available as easy command line standalone install.
- Includes XGBoost package (Linux* only)

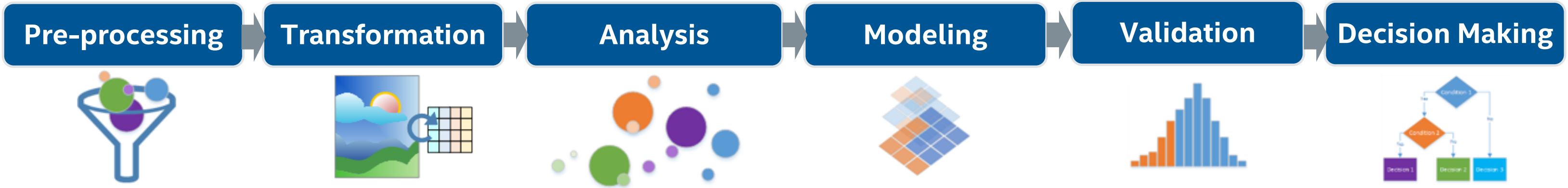
Software & workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark & MobileMark, are measured using specific computer systems, components, software, operations & functions. Any change to any of those factors may cause the results to vary. You should consult other information & performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to http://www.intel.com/performance

[2] Paid versions only

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

INTEL® DATA ANALYTICS ACCELERATION LIBRARY (INTEL® DAAL)

Building blocks for all data analytics stages, including data preparation, data mining & machine learning



Pre-processing → Transformation → Analysis → Modeling → Validation → Decision Making

Common Python*, Java*, C++ APIs across all Intel hardware

Optimizes data ingestion & algorithmic compute together for highest performance

Supports offline, streaming & distributed usage models for a range of application needs

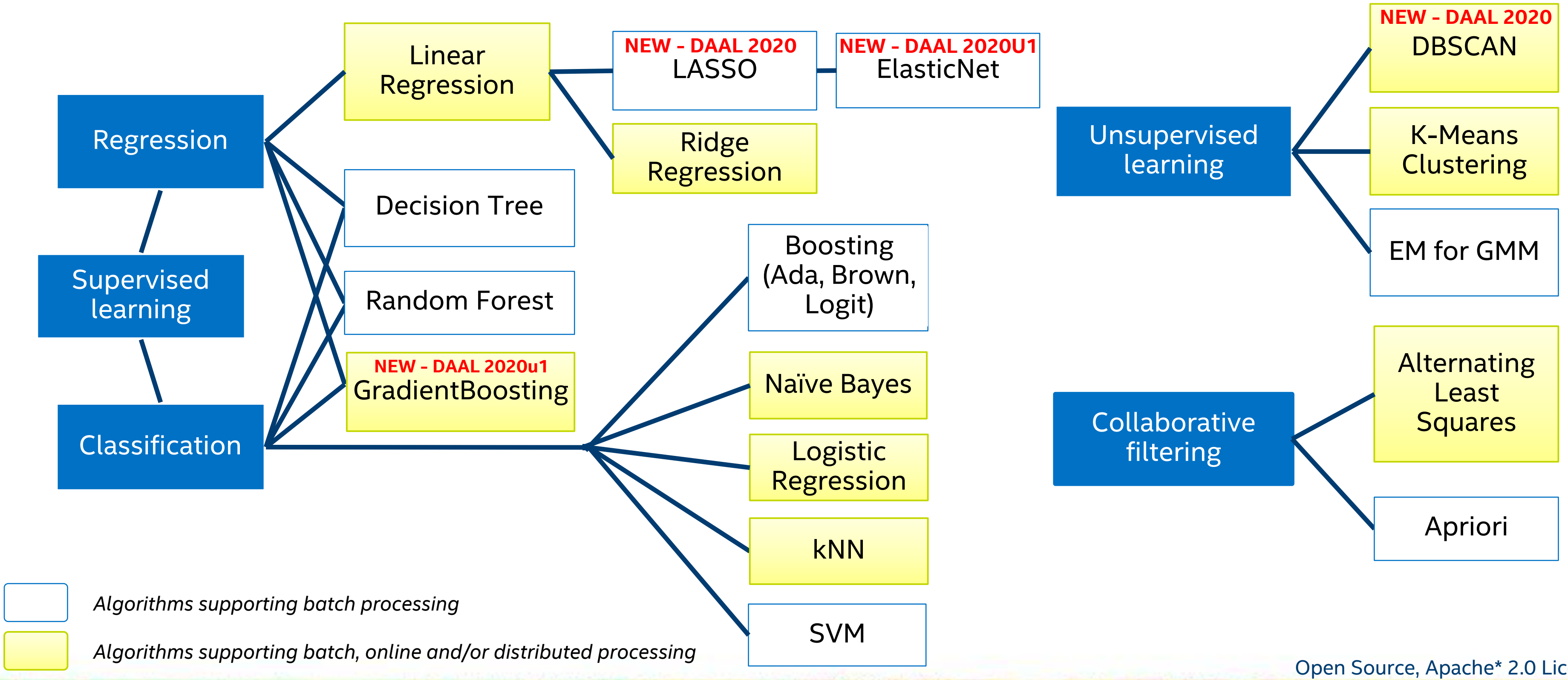
Flexible interfaces to leading big data platforms including Spark*

Split analytics workloads between edge devices & cloud to optimize overall application throughput

High Performance Machine Learning & Data Analytics Library

Open Source, Apache* 2.0 License

MACHINE LEARNING ALGORITHMS



Supervised learning

Regression

Linear Regression

NEW - DAAL 2020
LASSO

NEW - DAAL 2020U1
ElasticNet

Ridge Regression

Decision Tree

Random Forest

NEW - DAAL 2020u1
GradientBoosting

Classification

Boosting (Ada, Brown, Logit)

Naïve Bayes

Logistic Regression

kNN

SVM

Unsupervised learning

NEW - DAAL 2020
DBSCAN

K-Means Clustering

EM for GMM

Collaborative filtering

Alternating Least Squares

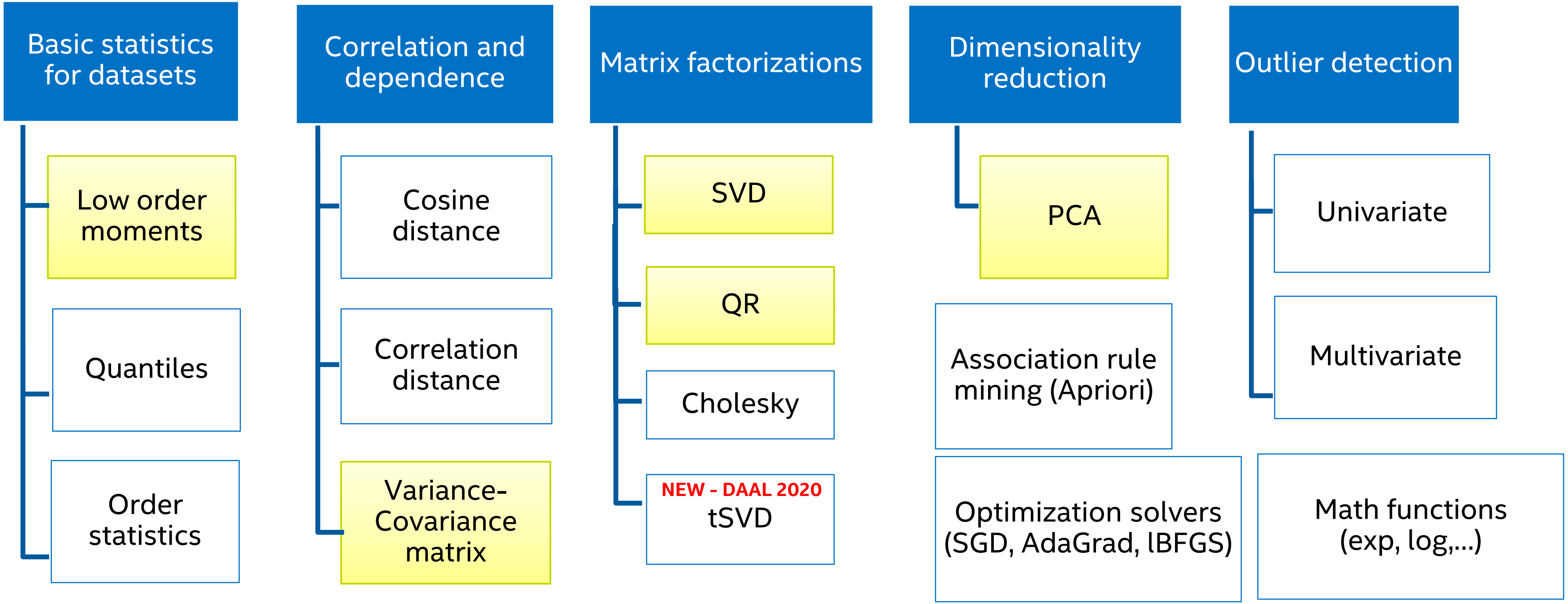
Apriori

□ *Algorithms supporting batch processing*

🟨 *Algorithms supporting batch, online and/or distributed processing*

Open Source, Apache* 2.0 License

DATA TRANSFORMATION & ANALYSIS ALGORITHMS



Basic statistics for datasets
- Low order moments
- Quantiles
- Order statistics

Correlation and dependence
- Cosine distance
- Correlation distance
- Variance-Covariance matrix

Matrix factorizations
- SVD
- QR
- Cholesky
- NEW - DAAL 2020 tSVD

Dimensionality reduction
- PCA

Association rule mining (Apriori)

Optimization solvers (SGD, AdaGrad, lBFGS)

Outlier detection
- Univariate
- Multivariate

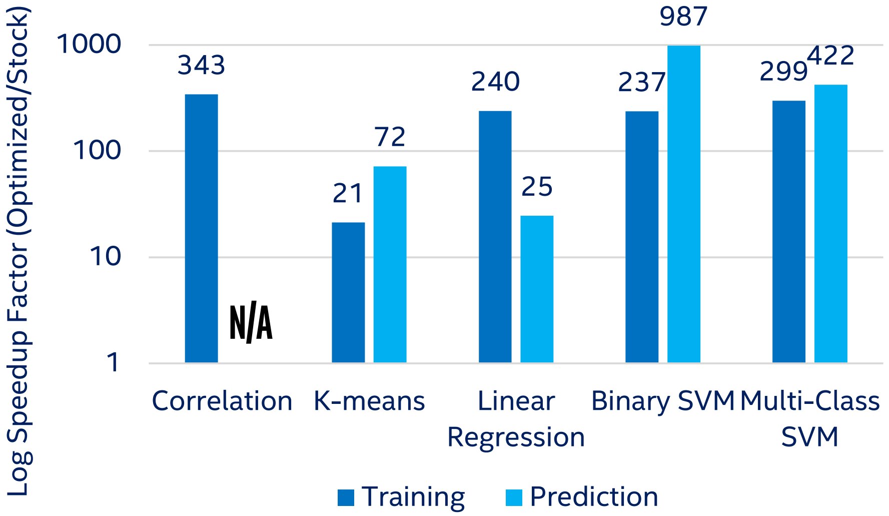
Math functions (exp, log,…)

□ *Algorithms supporting batch processing*

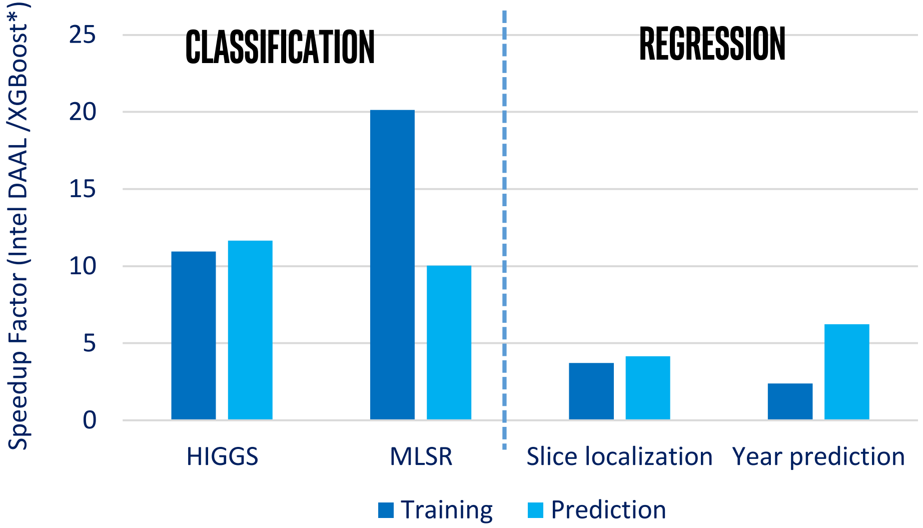
🟨 *Algorithms supporting batch, online and/or distributed processing*

INTEL® DAAL - PERFORMANCE

Intel® DAAL 2019 Log Scale Optimization of Scikit-learn*



Intel® Data Analytics Acceleration Library 2019 (Intel® DAAL) Speedup vs XGBoost*



Performance results are based on testing as of **July 9, 2018** and may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure.
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that proc when combined with other products. For more complete information, see Performance Benchmark Test Disclosure.

Testing by Intel as of July 9, 2018. Configuration: Intel® Xeon® Platinum 8180 H0 205W, 2x28@2.50GHz, 192GB, 12x16gb DDR4-2666, Intel® Data Analytics Acceleration Library (Intel® DAAL 2019), RHEL 7.2

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Inte not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimization specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804.
For more complete information about compiler optimizations, see our Optimization Notice.

Performance results are based on testing as of **July 9, 2018** and may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure.
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information, see Performance Benchmark Test Disclosure.

Testing by Intel as of July 9, 2018. Configuration: Intel® Xeon® Gold 6140 CPU, 2x18@2.30GHz, 256GB, 16x16gb DDR4-2666, Intel® Data Analytics Acceleration Library (Intel® DAAL 2019), Optimized: Scikit-learn*_intel 0.19.1, Numpy*_intel 1.14.3 Stock: Scikit-learn* 0.19.2, Numpy* 1.15.0, CentOS Linux 7.3.1611

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804
For more complete information about compiler optimizations, see our Optimization Notice.

FASTER, SCALABLE CODE WITH INTEL® MATH KERNEL LIBRARY

LINEAR ALGEBRA

- BLAS
- LAPACK
- ScaLAPACK
- Sparse BLAS
- Iterative sparse solvers
- PARDISO*
- Cluster Sparse Solver

FFTS

- Multidimensional
- FFTW interfaces
- Cluster FFT

VECTOR RNGS

- Congruential
- Wichmann-Hill
- Mersenne Twister
- Sobol
- Neirderreiter
- Non-deterministic

SUMMARY STATISTICS

- Kurtosis
- Variation coefficient
- Order statistics
- Min/max
- Variance-covariance

VECTOR MATH

- Trigonometric
- Hyperbolic
- Exponential
- Log
- Power
- Root

AND MORE

- Splines
- Interpolation
- Trust Region
- Fast Poisson Solver

INTEL® MATH KERNEL LIBRARY FOR DEEP NEURAL NETWORKS (INTEL® MKL-DNN)

For developers of deep learning frameworks featuring optimized performance on Intel hardware

Distribution Details

- Open Source
- Apache* 2.0 License
- Common DNN APIs across all Intel hardware.
- Rapid release cycles, iterated with the DL community, to best support industry framework integration.
- Highly vectorized & threaded for maximal performance, based on the popular Intel® Math Kernel Library.

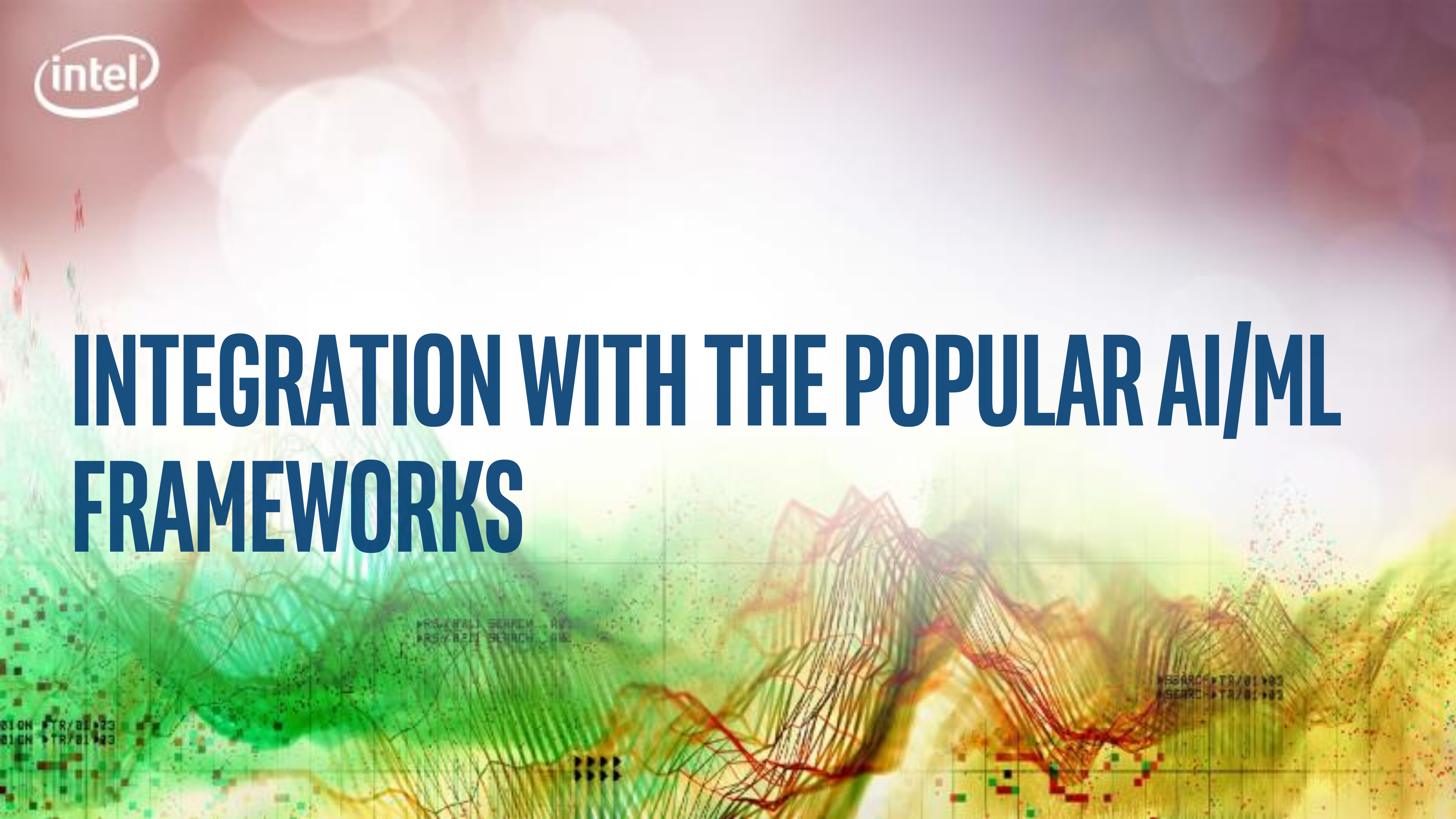
github.com/01org/mkl-dnn

Examples:

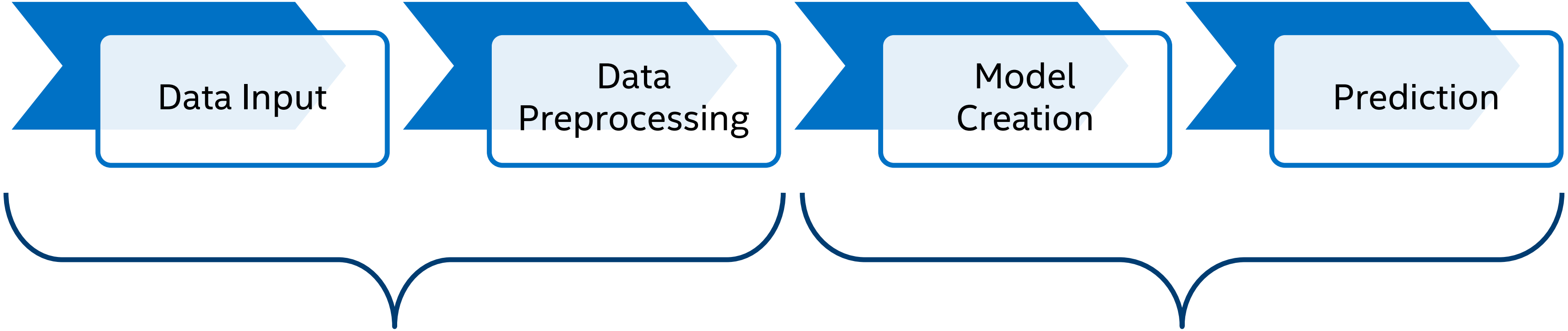
Direct 2D Convolution
Local response normalization (LRN)
Rectified linear unit neuron activation (ReLU)
Maximum pooling
Inner product

Accelerate Performance of Deep Learning Models

INTEGRATION WITH THE POPULAR AI/ML FRAMEWORKS

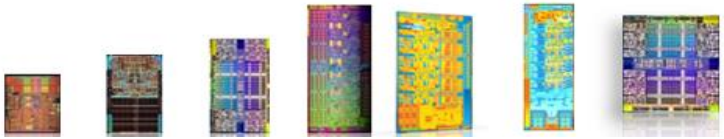


DATA ANALYSIS AND MACHINE LEARNING



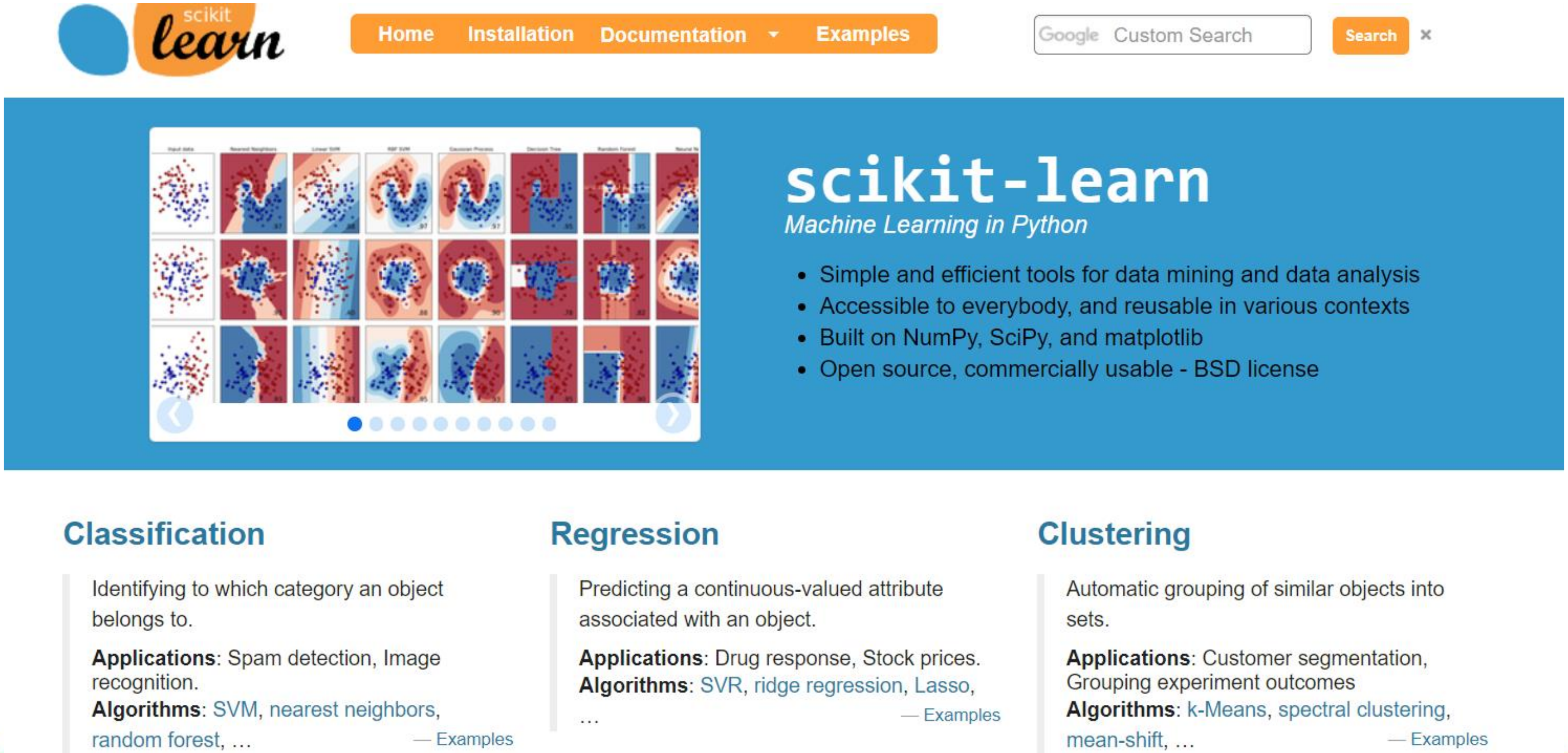
Pandas
Spark
HPAT

Scikit-learn
Spark
DL-frameworks
daal4py



more nodes, more cores, more threads, wider vectors, …

THE MOST POPULAR ML PACKAGE FOR PYTHON*



scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, …

— Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, …

— Examples

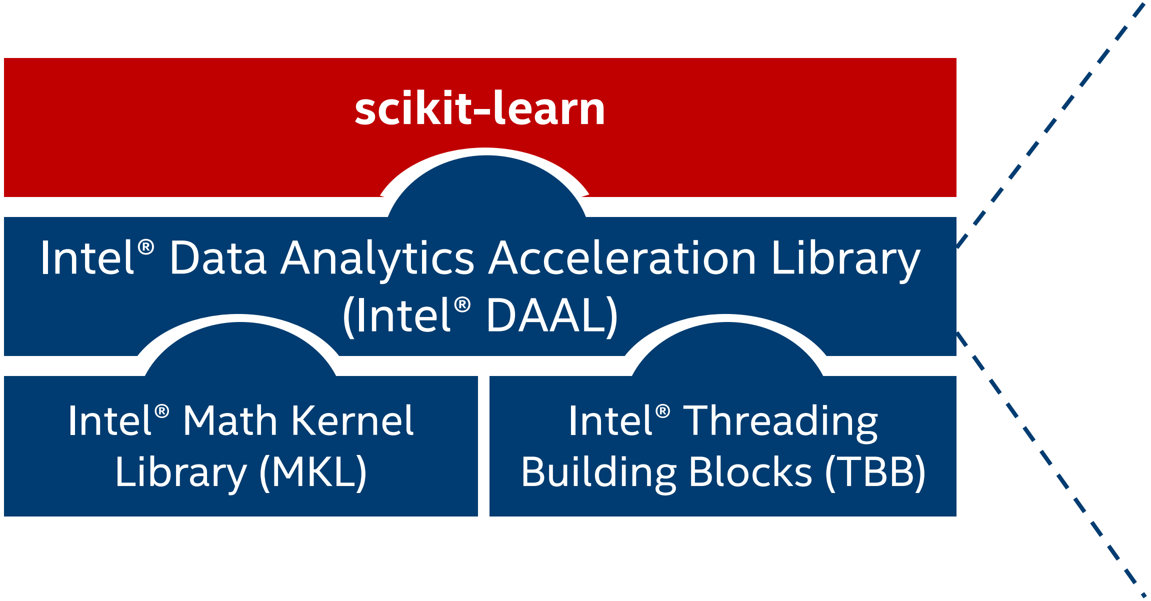
Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, …

— Examples

ACCELERATING MACHINE LEARNING



scikit-learn

Intel® Data Analytics Acceleration Library (Intel® DAAL)

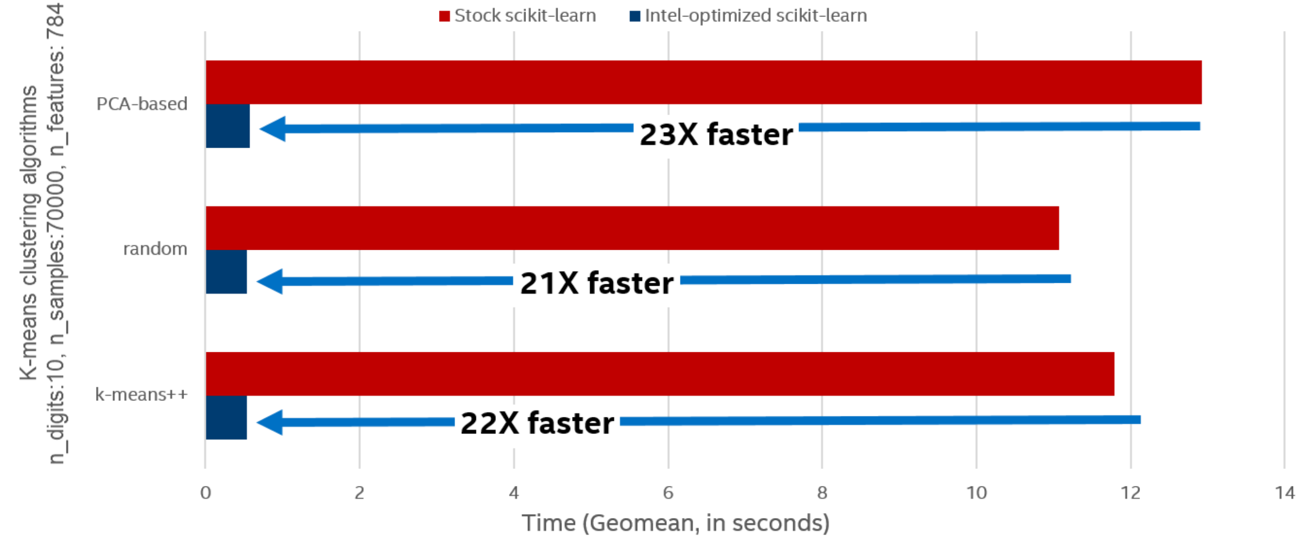
Intel® Math Kernel Library (MKL)

Intel® Threading Building Blocks (TBB)

- **Efficient memory layout** via `Numeric Tables`
- **Blocking** for optimal cache performance
- **Computation mapped to most efficient matrix operations** (in MKL)
- **Parallelization** via TBB
- **Vectorization**

Try it out! `conda install -c intel scikit-learn`

ACCELERATING K-MEANS



Performance speedups for Intel® Distribution for Python* scikit-learn on Google Cloud Platform's 96 vCPU instance Intel® Xeon™ Processors

System Configuration: GCP VM, zone us-central1-c; 96 vCPU, Intel Skylake; 360 GB memory. Ubuntu 16.04.3 LTS; Linux instance-1 4.10.0-38-generic #42~16.04.1-Ubuntu SMP Tue Oct 10 16:32:20 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux; Intel® Distribution for Python* from Docker image intelpython/intelpython3_full:latest (created 2017-09-12T20:10:42.862965559Z); Stock Python*: pip install scikit-learn

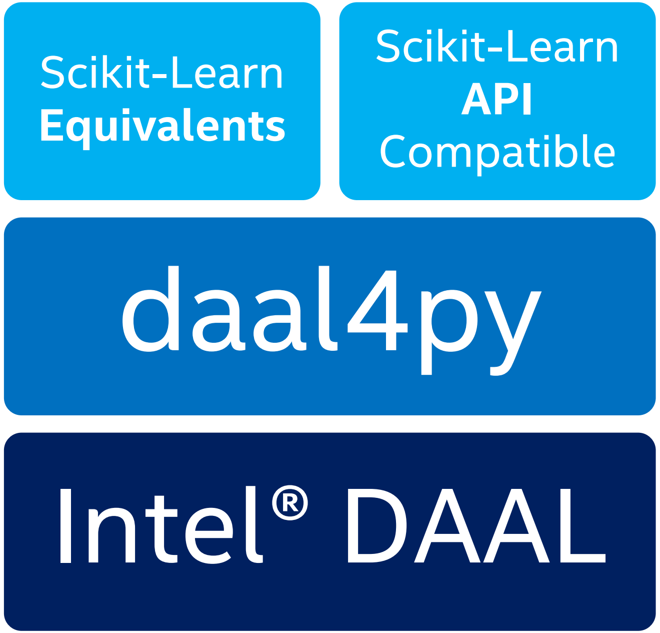
https://cloudplatform.googleblog.com/2017/11/Intel-performance-libraries-and-python-distribution-enhance-performance-and-scaling-of-Intel-Xeon-Scalable-processors-on-GCP.html

DAAL4PY

Fast & Scalable	• Close to native performance through Intel® DAAL • Efficient MPI scale-out • Streaming
Easy to use	• Known usage model • Picklable
Flexible	• Object model separating concerns • Plugs into scikit-learn • Plugs into HPAT
Open	• Open source: https://github.com/IntelPython/daal4py

https://intelpython.github.io/daal4py/

ACCELERATING SCIKIT-LEARN THROUGH DAAL4PY



```
> python -m daal4py <your-scikit-learn-script>
```

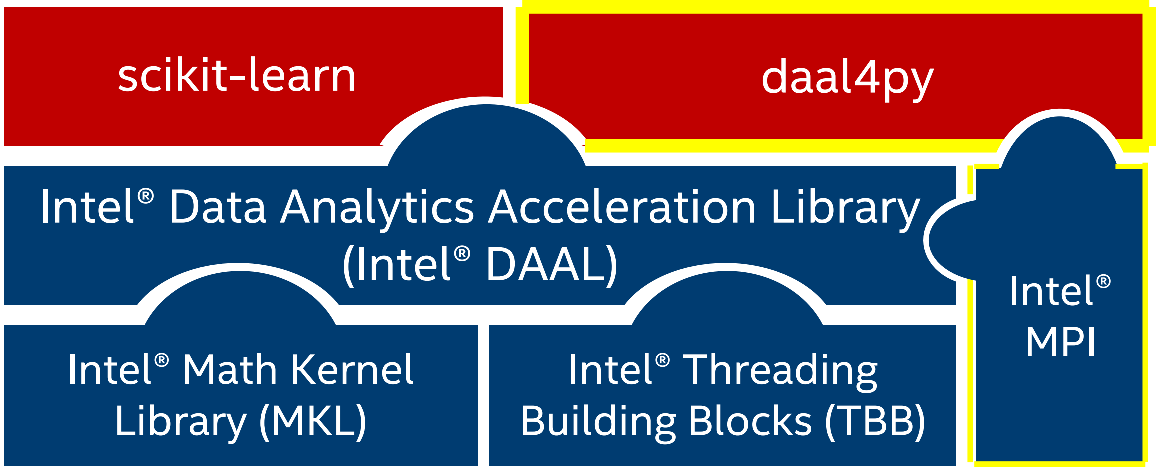
Monkey-patch any scikit-learn on the command-line

```
import daal4py.sklearn
daal4py.sklearn.patch_sklearn()
```

Monkey-patch any scikit-learn programmatically

https://intelpython.github.io/daal4py/

SCALING MACHINE LEARNING BEYOND A SINGLE NODE



scikit-learn

daal4py

Intel® Data Analytics Acceleration Library (Intel® DAAL)

Intel® Math Kernel Library (MKL)

Intel® Threading Building Blocks (TBB)

Intel® MPI

Simple Python API
Powers scikit-learn

Powered by DAAL

Scalable to multiple nodes

Try it out! `conda install -c intel daal4py`

K-MEANS USING DAAL4PY

```
import daal4py as d4p

# daal4py accepts data as CSV files, numpy arrays or pandas dataframes
# here we let daal4py load process-local data from csv files
data = "kmeans_dense.csv"

# Create algob object to compute initial centers
init = d4p.kmeans_init(10, method="plusPlusDense")
# compute initial centers
ires = init.compute(data)
# results can have multiple attributes, we need centroids
Centroids = ires.centroids
# compute initial centroids & kmeans clustering
result = d4p.kmeans(10).compute(data, centroids)
```

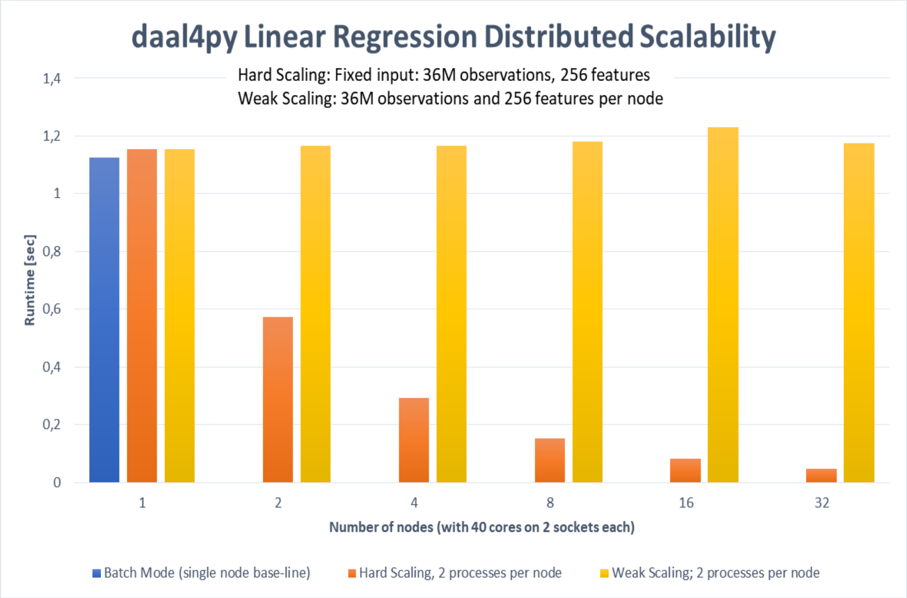
DISTRIBUTED K-MEANS USING DAAL4PY

```
import daal4py as d4p

# initialize distributed execution environment
d4p.daalinit()

# daal4py accepts data as CSV files, numpy arrays or pandas dataframes
# here we let daal4py load process-local data from csv files
data = "kmeans_dense_{}.csv".format(d4p.my_procid())

# compute initial centroids & kmeans clustering
init = d4p.kmeans_init(10, method="plusPlusDense", distributed=True)
centroids = init.compute(data).centroids
result = d4p.kmeans(10, distributed=True).compute(data, centroids)
```

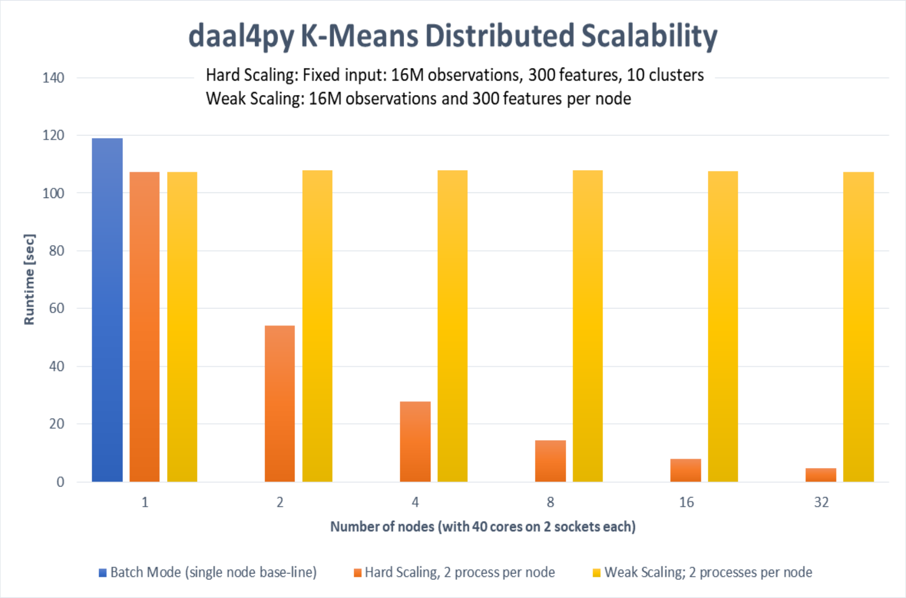
```
mpirun -n 4 python ./kmeans.py
```

STRONG & WEAK SCALING VIA DAAL4PY

Hardware	Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz, EIST/Turbo on
	2 sockets, 20 Cores per socket
	192 GB RAM
	16 nodes connected with Infiniband
Operating System	Oracle Linux Server release 7.4
Data Type	double



On a 32-node cluster (1280 cores) daal4py computed linear regression of 2.15 TB of data in 1.18 seconds and 68.66 GB of data in less than 48 milliseconds.



On a 32-node cluster (1280 cores) daal4py computed K-Means (10 clusters) of 1.12 TB of data in 107.4 seconds and 35.76 GB of data in 4.8 seconds.

intel®

experience
what's inside™