# AI frameworks

# TensorFlow

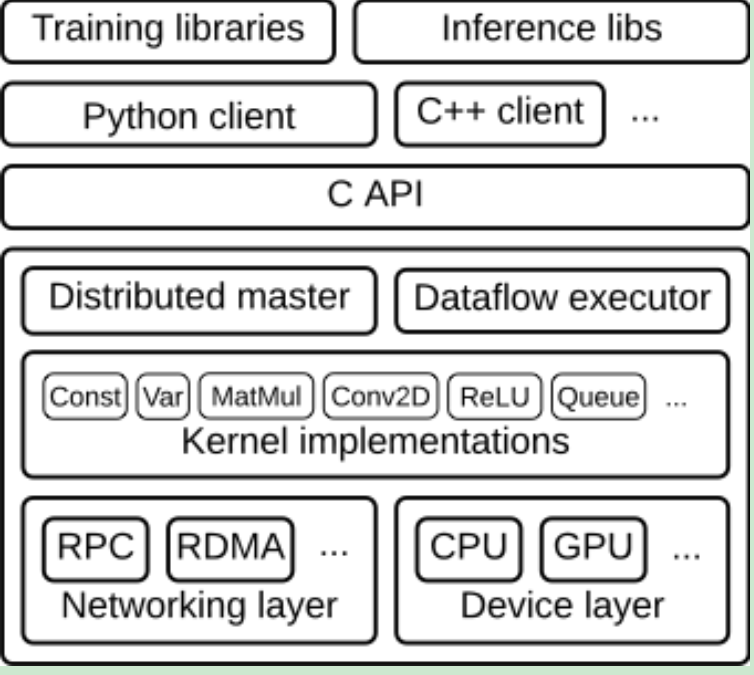
## 1.1.TensorFlow简介：

TensorFlow™ is an open source software library for high performance numerical computation. Its flexible architecture allows easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.

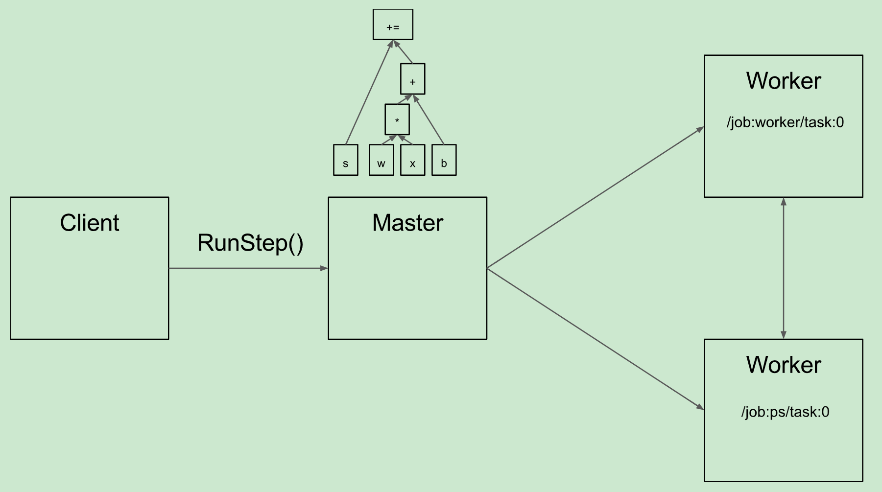
TensorFlow是谷歌在开发的2.0版本的深度学习框架，最初主要针对深度学习模型框架开发，现已支持部分机器学习模型。对GPU,TPU, 分布式模型训练及应用支持。

## 1.2.TensorFlow架构：

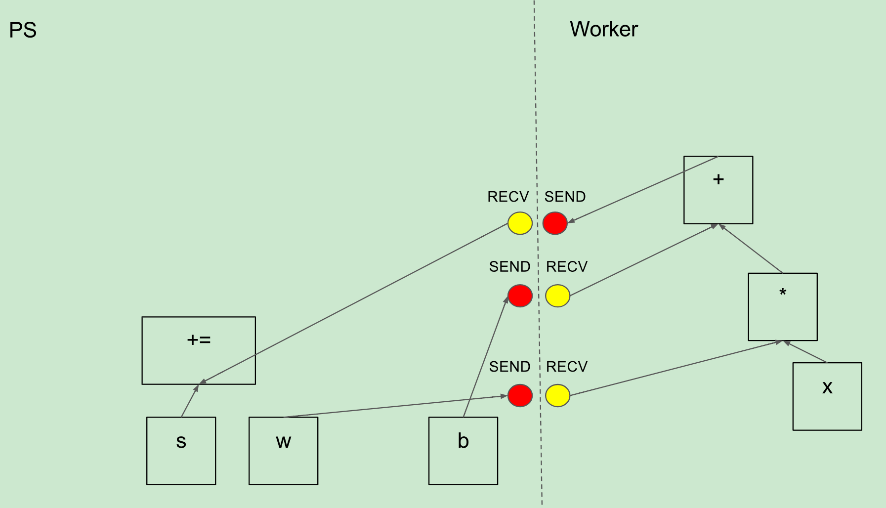
整体架构

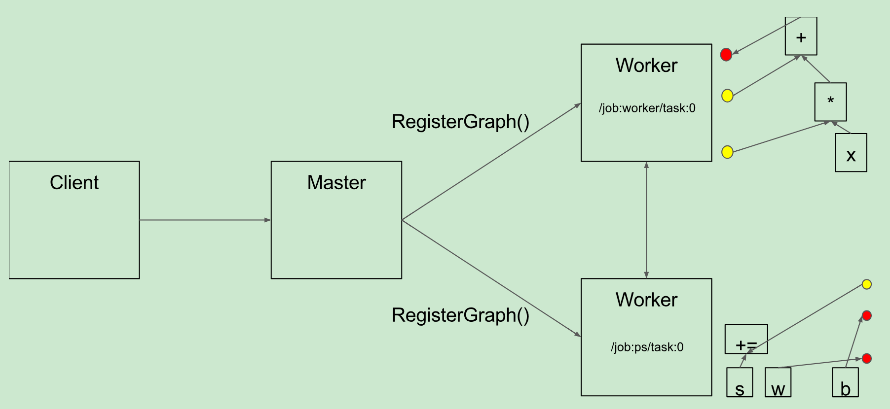


分布式架构



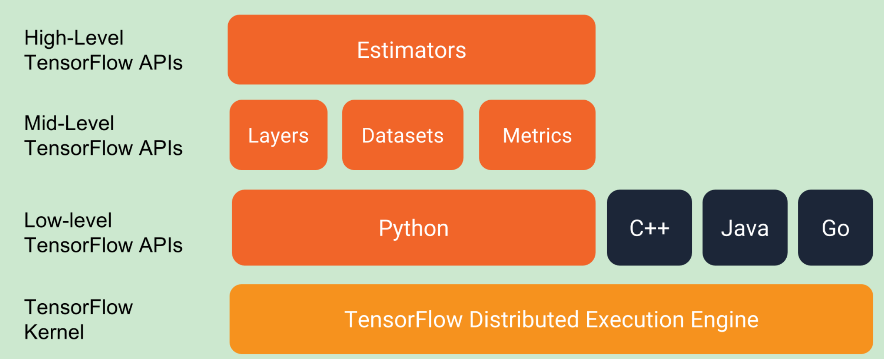
分布式通讯





## 1.3.TensorFlow API

TensorFlow现功能最全最稳定的API为pyhon版本，现最新版为1.11.



## 1.4.TensorFlow支持算法

TensorFlow支持High-level算法支持：

[BoostedTreesClassifier](https://www.tensorflow.org/api_docs/python/tf/estimator/BoostedTreesClassifier): A Classifier for Tensorflow Boosted Trees models.

[BoostedTreesRegressor](https://www.tensorflow.org/api_docs/python/tf/estimator/BoostedTreesRegressor): A Regressor for Tensorflow Boosted Trees models.

[DNNClassifier](https://www.tensorflow.org/api_docs/python/tf/estimator/DNNClassifier): A classifier for TensorFlow DNN models.

[DNNLinearCombinedClassifier](https://www.tensorflow.org/api_docs/python/tf/estimator/DNNLinearCombinedClassifier): An estimator for TensorFlow Linear and DNN joined classification models.

[DNNLinearCombinedRegressor](https://www.tensorflow.org/api_docs/python/tf/estimator/DNNLinearCombinedRegressor): An estimator for TensorFlow Linear and DNN joined models for regression.

[DNNRegressor](https://www.tensorflow.org/api_docs/python/tf/estimator/DNNRegressor): A regressor for TensorFlow DNN models.

[LinearClassifier](https://www.tensorflow.org/api_docs/python/tf/estimator/LinearClassifier): Linear classifier model.

[LinearRegressor](https://www.tensorflow.org/api_docs/python/tf/estimator/LinearRegressor): An estimator for TensorFlow Linear regression problems.

TensorFlow Low-level 算法支持：

可以利用Low-level API 开发各类深度学习算法，如：DNN,CNN, RNN, LSTM, Auto-Encoder, GAN等

## 1.5.TensorFlow的优势：

1.灵活；2.轻便；3.科研及生产；4.自微分；5.性能

**1. Flexibility:** You need to express your computation as a data flow graph to use TensorFlow. It is a highly flexible system which provides multiple models or multiple versions of the same model can be served simultaneously. The architecture of TensorFlow is highly modular, which means you can use some parts individually or can use all the parts together. Such flexibility facilitates non-automatic migration to new models/versions, A/B testing experimental models, and canarying new models.

**2.** **Portability:** TensorFlow has made it possible to play around an idea on your laptop without having any other hardware support. It runs on **GPUs, CPUs, desktops, servers**, and **mobile** computing platforms. You can deploy a trained model on your mobile as a part of your product, and that’s how it serves as a true portability feature.

**3. Research and Production:** It can be used to train and serve models in live mode to real customers. To put it simply, rewriting codes is not required and the industrial researchers can apply their ideas to products faster. Also, academic researchers can share codes directly with greater **reproducibility**. In this way it helps to carry out research and **production** processes faster.

**4. Auto Differentiation:** It has automatic differentiation capabilities which benefits gradient based machine learning algorithms. You can define the computational architecture of your predictive model, combine it with your objective function and add data to it- TensorFlow manages **derivatives** computing processes automatically. You can compute the derivatives of some values with respect to some other values results in graph extension and you can see exactly what’s happening.

**5. Performance:** TensorFlow allows you to make the most of your **available hardware** with its advanced support for **threads, asynchronous computation, and queues**. Just assign compute elements of your TensorFlow graph to different devices and let it manage the copies itself. It also facilitates you with the language options to execute your computational graph. TensorFlow iPython notebook helps in keeping codes, notes, and visualization in a logically grouped and **interactive** style.

## 1.6.TensorFlow应用场景

1. Voice/Sound Recognition;2. Text-Based Applications;3.Image Recognition;4.Time Series;5.Video Detection;6.NLP

对于大部分的深度学习应用领域，TensorFlow绝大部分都能够支持模型开发。

## 1.7.TensorFlow serving

TensorFlow Serving is a **flexible, high-performance** serving system for machine learning models, designed for **production** environments. TensorFlow Serving makes it easy to deploy new algorithms and experiments, while keeping the same server architecture and APIs. TensorFlow Serving provides out-of-the-box integration with TensorFlow models, but can be easily extended to serve other types of models and data.

# SparkMLlib

## 2.1.SparkMLlib简介

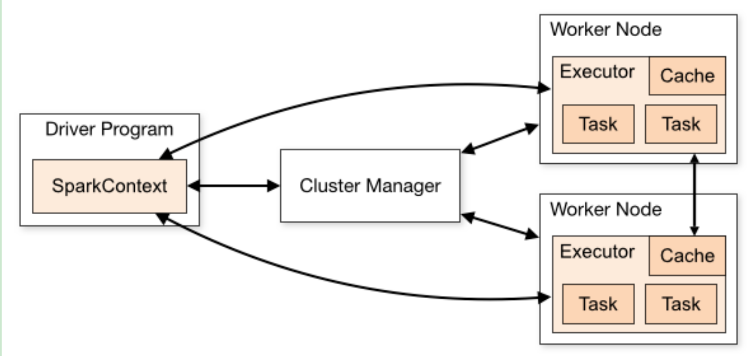
MLlib is Spark’s machine learning (ML) library. Its goal is to make practical machine learning scalable and easy. At a high level, it provides tools such as:

* ML Algorithms: common learning algorithms such as classification, regression, clustering, and collaborative filtering
* Featurization: feature extraction, transformation, dimensionality reduction, and selection
* Pipelines: tools for constructing, evaluating, and tuning ML Pipelines
* Persistence: saving and load algorithms, models, and Pipelines
* Utilities: linear algebra, statistics, data handling, etc.

SparkMLlib主要是针对大数据机器学习模型开发及上线应用的大数据框架，对传统的机器学习支持较好，对深度学习不支持。

## 2.2.Spark架构

Spark applications run as independent sets of processes on a cluster, coordinated by the SparkContext object in your main program (called the driver program).Specifically, to run on a cluster, the SparkContext can connect to several types of cluster managers (either Spark’s own standalone cluster manager, Mesos or YARN), which allocate resources across applications. Once connected, Spark acquires executors on nodes in the cluster, which are processes that run computations and store data for your application. Next, it sends your application code (defined by JAR or Python files passed to SparkContext) to the executors. Finally, SparkContext sends tasks to the executors to run.



## 2.3.MLlib支持算法

从特征工程算法 ，分类回归聚类算法，pipeline，模型选择交叉验证，模型评估等均有较好支持。

特征工程算法：

* [Feature Extractors](https://spark.apache.org/docs/latest/ml-features.html#feature-extractors)
  + [TF-IDF](https://spark.apache.org/docs/latest/ml-features.html#tf-idf)
  + [Word2Vec](https://spark.apache.org/docs/latest/ml-features.html#word2vec)
  + [CountVectorizer](https://spark.apache.org/docs/latest/ml-features.html#countvectorizer)
  + [FeatureHasher](https://spark.apache.org/docs/latest/ml-features.html#featurehasher)
* [Feature Transformers](https://spark.apache.org/docs/latest/ml-features.html#feature-transformers)
  + [Tokenizer](https://spark.apache.org/docs/latest/ml-features.html#tokenizer)
  + [StopWordsRemover](https://spark.apache.org/docs/latest/ml-features.html#stopwordsremover)
  + [nn-gram](https://spark.apache.org/docs/latest/ml-features.html#n-gram)
  + [Binarizer](https://spark.apache.org/docs/latest/ml-features.html#binarizer)
  + [PCA](https://spark.apache.org/docs/latest/ml-features.html#pca)
  + [PolynomialExpansion](https://spark.apache.org/docs/latest/ml-features.html#polynomialexpansion)
  + [Discrete Cosine Transform (DCT)](https://spark.apache.org/docs/latest/ml-features.html#discrete-cosine-transform-dct)
  + [StringIndexer](https://spark.apache.org/docs/latest/ml-features.html#stringindexer)
  + [IndexToString](https://spark.apache.org/docs/latest/ml-features.html#indextostring)
  + [OneHotEncoder (Deprecated since 2.3.0)](https://spark.apache.org/docs/latest/ml-features.html#onehotencoder-deprecated-since-230)
  + [OneHotEncoderEstimator](https://spark.apache.org/docs/latest/ml-features.html#onehotencoderestimator)
  + [VectorIndexer](https://spark.apache.org/docs/latest/ml-features.html#vectorindexer)
  + [Interaction](https://spark.apache.org/docs/latest/ml-features.html#interaction)
  + [Normalizer](https://spark.apache.org/docs/latest/ml-features.html#normalizer)
  + [StandardScaler](https://spark.apache.org/docs/latest/ml-features.html#standardscaler)
  + [MinMaxScaler](https://spark.apache.org/docs/latest/ml-features.html#minmaxscaler)
  + [MaxAbsScaler](https://spark.apache.org/docs/latest/ml-features.html#maxabsscaler)
  + [Bucketizer](https://spark.apache.org/docs/latest/ml-features.html#bucketizer)
  + [ElementwiseProduct](https://spark.apache.org/docs/latest/ml-features.html#elementwiseproduct)
  + [SQLTransformer](https://spark.apache.org/docs/latest/ml-features.html#sqltransformer)
  + [VectorAssembler](https://spark.apache.org/docs/latest/ml-features.html#vectorassembler)
  + [VectorSizeHint](https://spark.apache.org/docs/latest/ml-features.html#vectorsizehint)
  + [QuantileDiscretizer](https://spark.apache.org/docs/latest/ml-features.html#quantilediscretizer)
  + [Imputer](https://spark.apache.org/docs/latest/ml-features.html#imputer)
* [Feature Selectors](https://spark.apache.org/docs/latest/ml-features.html#feature-selectors)
  + [VectorSlicer](https://spark.apache.org/docs/latest/ml-features.html#vectorslicer)
  + [RFormula](https://spark.apache.org/docs/latest/ml-features.html#rformula)
  + [ChiSqSelector](https://spark.apache.org/docs/latest/ml-features.html#chisqselector)

分类算法：

[Classification](https://spark.apache.org/docs/latest/ml-classification-regression.html#classification)

* [Logistic regression](https://spark.apache.org/docs/latest/ml-classification-regression.html#logistic-regression)
  + [Binomial logistic regression](https://spark.apache.org/docs/latest/ml-classification-regression.html#binomial-logistic-regression)
  + [Multinomial logistic regression](https://spark.apache.org/docs/latest/ml-classification-regression.html#multinomial-logistic-regression)
* [Decision tree classifier](https://spark.apache.org/docs/latest/ml-classification-regression.html#decision-tree-classifier)
* [Random forest classifier](https://spark.apache.org/docs/latest/ml-classification-regression.html#random-forest-classifier)
* [Gradient-boosted tree classifier](https://spark.apache.org/docs/latest/ml-classification-regression.html#gradient-boosted-tree-classifier)
* [Multilayer perceptron classifier](https://spark.apache.org/docs/latest/ml-classification-regression.html#multilayer-perceptron-classifier)
* [Linear Support Vector Machine](https://spark.apache.org/docs/latest/ml-classification-regression.html#linear-support-vector-machine)
* [One-vs-Rest classifier (a.k.a. One-vs-All)](https://spark.apache.org/docs/latest/ml-classification-regression.html#one-vs-rest-classifier-aka-one-vs-all)
* [Naive Bayes](https://spark.apache.org/docs/latest/ml-classification-regression.html#naive-bayes)

回归算法：

[Regression](https://spark.apache.org/docs/latest/ml-classification-regression.html#regression)

* [Linear regression](https://spark.apache.org/docs/latest/ml-classification-regression.html#linear-regression)
* [Generalized linear regression](https://spark.apache.org/docs/latest/ml-classification-regression.html#generalized-linear-regression)
  + [Available families](https://spark.apache.org/docs/latest/ml-classification-regression.html#available-families)
* [Decision tree regression](https://spark.apache.org/docs/latest/ml-classification-regression.html#decision-tree-regression)
* [Random forest regression](https://spark.apache.org/docs/latest/ml-classification-regression.html#random-forest-regression)
* [Gradient-boosted tree regression](https://spark.apache.org/docs/latest/ml-classification-regression.html#gradient-boosted-tree-regression)
* [Survival regression](https://spark.apache.org/docs/latest/ml-classification-regression.html#survival-regression)
* [Isotonic regression](https://spark.apache.org/docs/latest/ml-classification-regression.html#isotonic-regression)

聚类算法：

* [K-means](https://spark.apache.org/docs/latest/ml-clustering.html#k-means)
  + [Input Columns](https://spark.apache.org/docs/latest/ml-clustering.html#input-columns)
  + [Output Columns](https://spark.apache.org/docs/latest/ml-clustering.html#output-columns)
* [Latent Dirichlet allocation (LDA)](https://spark.apache.org/docs/latest/ml-clustering.html#latent-dirichlet-allocation-lda)
* [Bisecting k-means](https://spark.apache.org/docs/latest/ml-clustering.html#bisecting-k-means)
* [Gaussian Mixture Model (GMM)](https://spark.apache.org/docs/latest/ml-clustering.html#gaussian-mixture-model-gmm)
  + [Input Columns](https://spark.apache.org/docs/latest/ml-clustering.html#input-columns-1)
  + [Output Columns](https://spark.apache.org/docs/latest/ml-clustering.html#output-columns-1)

## 2.4.MLlib优势

SparkMLlib主要是基于大数据平台Spark的大数据机器学习框架，对机器学习算法、特征工程算法、模型选择算法都有很好的支持。

不过对一部分的机器学习算法不支持：如SVM, RidgeRegression等算法不支持，而且对深度学习算法不支持。

# Scikit-learn

## 3.1.SKlearn简介

## 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

SKlearn是python机器学习最全的工具，对于传统的机器学习算法和特征工程算法都有支持。

## 3.2.SKlearn支持算法

SKlearn是对机器学习算法的支持非常多，而且主要大部分的算法都支持，对大部分的数据预处理算法也都有支持。

[**Generalized Linear Models**](http://scikit-learn.org/stable/modules/linear_model.html)

[**Linear and Quadratic Discriminant Analysis**](http://scikit-learn.org/stable/modules/lda_qda.html)

[**Kernel ridge regression**](http://scikit-learn.org/stable/modules/kernel_ridge.html)

[**Support Vector Machines**](http://scikit-learn.org/stable/modules/svm.html)

[**Stochastic Gradient Descent**](http://scikit-learn.org/stable/modules/sgd.html)

[**Nearest Neighbors**](http://scikit-learn.org/stable/modules/neighbors.html)

[**Gaussian Processes**](http://scikit-learn.org/stable/modules/gaussian_process.html)

[**Cross decomposition**](http://scikit-learn.org/stable/modules/cross_decomposition.html)

[**Naive Bayes**](http://scikit-learn.org/stable/modules/naive_bayes.html)

[**Decision Trees**](http://scikit-learn.org/stable/modules/tree.html)

[**Ensemble methods**](http://scikit-learn.org/stable/modules/ensemble.html)

[**Multiclass and multilabel algorithms**](http://scikit-learn.org/stable/modules/multiclass.html)

[**Feature selection**](http://scikit-learn.org/stable/modules/feature_selection.html)

[**Neural network models (supervised)**](http://scikit-learn.org/stable/modules/neural_networks_supervised.html)

## 3.3.SKlearn的优势：

SKlearn对绝大部分的特征工程算法，机器学习算法，模型选择算法，模型评估算法都支持的比较好，在小数据量上利用机器学习算法效果较好；

不过sklearn对深度学习的算法不支持，大数据量的效果没有深度学习算法效果好。