Logistic Regression Core

BGD IP core for logistic regression acceleration

Logistic regression

Logistic regression is used for building predictive models for many complex pattern-matching and classification problems. It is used widely in such diverse areas as bioinformatics, finance and data analytics. It is also one of the most popular machine learning techniques. It belongs to the family of classifiers known as the exponential or log-linear classifiers and is widely used to predict a binary response.

For multi-class classification problems, the algorithm compares every class with all the remaining classes (One versus Rest) and outputs a multinomial logistic regression model, which contains numClasses (k) binary logistic regression models. Given a new data point, k models will be run, and the class with largest probability will be chosen as the predicted class.

The specific IP core implements the (Batch) Gradient Descent algorithm for the Logistic Regression.

Logistic regression IP core

Features:

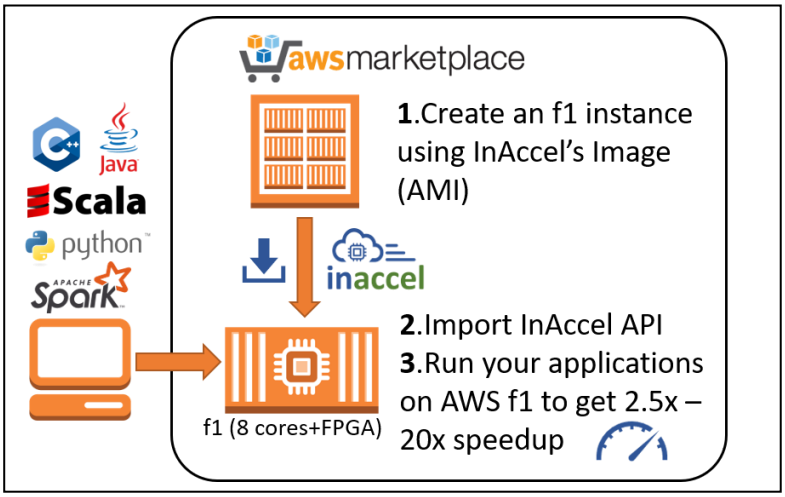
- Up to 25x speedup compared to single-thread contemporary processors
- Up to 2.4x speedup compared to 36-core contemporary processors
- APIs for C/C++, Python, Java, and Scala
- APIs and libraries for Spark integration

Spark Integration

The LR IP cores is compatible with the Spark ML lib on logistic regression.

In Spark gradients kernel can be parallelized using Map-Reduce, so partial gradients are computed in each Worker, using different chunks of the training set, and then the Master aggregates them and updates w.

The Spark code for the utilization of the hardware accelerator through our accelerated machine learning library is shown in the figure. When the Spark user wants to utilize the hardware accelerator, the only change that needs to be made is the replacement of the Spark mllib library with the mllib_accel library. Therefore, the user can speedup the execution time of the Spark application with a simple replacement of the libraries that wish to accelerate.




```
from pyspark import SparkContext
from pyspark.mllib.regression import LabeledPoint
from mllib_accel.classification import
LogisticRegression
```