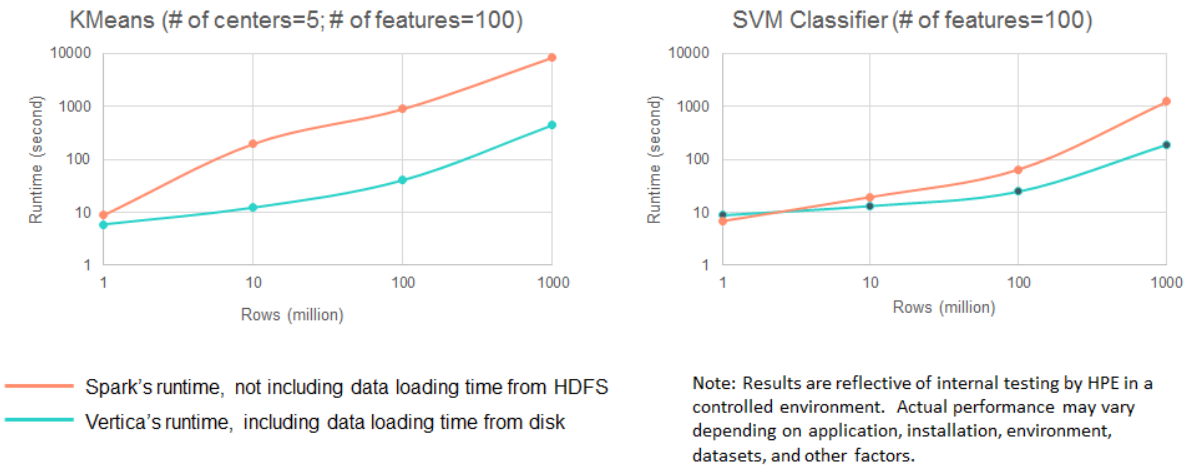
# Machine Learning Mondays: Vertica における効率的でスケーラブルな機械学習の実装方法

原文はこちら

Vertica 8.1 までに、Vertica は、線形回帰、ロジスティック回帰、Kmeans、Naive Bayes、SVM など、一般的な機械学習のアルゴリズムを導入しました。最近のベンチマークに基づくと、Apache Spark の MLlib よりも高速に動作します。次の図は、Vertica 8.1.0 と Spark 2.1.0 の間のパフォーマンスの違いを示しています（軸に表示されている数値は対数スケールです）。



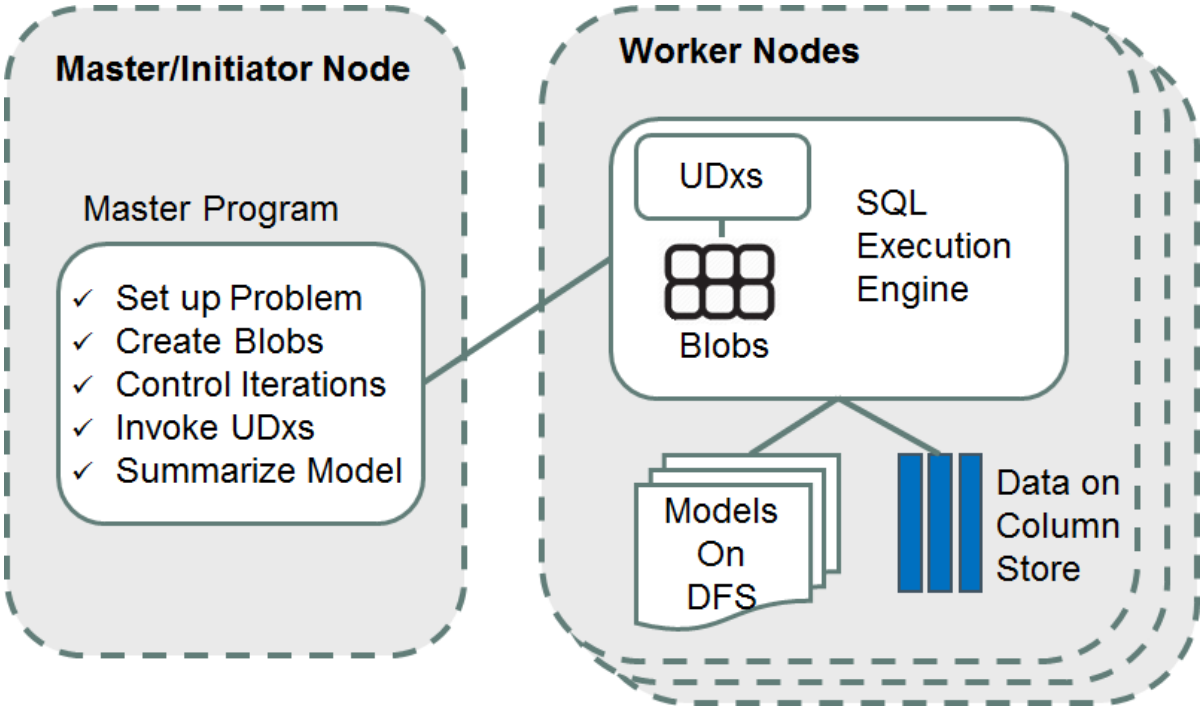

Vertica の機械学習は、サンプルサイズ、機能の数、およびクラスタサイズに沿ってスケーラブルです。何よりも、ダウンサンプリングは必要ありません。Vertica の機械学習が処理できるデータのサイズは、Vertica が保存できるデータのサイズにのみ制限されます。そのような成果は偶然ではありません。Vertica の強力な SQL エンジンと最先端の分散コンピューティングフレームワークの両方を活用するバランスの取れたアーキテクチャを設計するために、多くの反復作業を行ってきました。

機械学習機能を RDBMS に統合することの価値は、長い間認識されてきました。データが存在するデータベースで機械学習アルゴリズムを実行すると、データの移動が最小限に抑えられ、機械学習ワークフローが大幅に簡素化されます。ほとんどの RDBMS ベンダーは、機械学習を組み込むさまざまなアプローチを試してきました。従来のデータベースベンダーは、集中管理されたシステムで機械学習機能を開発しました。MADlib は、SQL エンジン上で機械学習を実行しようとします。しかしながら、これらの取り組みはスケーラビリティにおいて同様の課題を抱えています。最近では、インメモリ分散型システムは、機械学習を含む大規模な分析のための一般的なプラットフォームとなっています。Apache Spark などのこれらのプラットフォームは、コンピューティングエンジンとしてのみ機能し、永続データストレージは Hadoop などの他のシステムとの統合によって提供されます。

Vertica の目標は、スケーラブルな機械学習もサポートするスケーラブルなデータベースを提供することです。Vertica の機械学習の設計は、Vertica の分散プラットフォームを活用し、インメモリ処理を採用し、SQL エンジンと問題なく共存します。次の記事では、設計の重要な技術について説明します。

## マスター/ワーカーフレームワーク

次の図に示すように、Vertica の機械学習は Vertica の分散インフラストラクチャを活用し、マスター/ワーカーフレームワークを採用しています。どのノードでも実行可能なマスタープログラムは、アルゴリズムの高水準なワークフローを制御します。問題を設定し、Vertica のユーザー定義関数（UDx）を使用してローカルデータに基づいて学習の反復を行うように各ワーカーに指示します。マスターがアルゴリズムが収束したと判断すると、各ノードからのすべての学習を統合し、モデルを出力します。




Vertica 上のデータは分散され、機械学習アルゴリズムは分散の利点を利用し、ノード上でローカルにあるデータの重い計算を実行して、ノード間のデータ移動を最小限に抑えます。

## インメモリ処理

Vertica は MPP アーキテクチャで構築されています。その実行エンジンは、SQL クエリのようなワンパスアルゴリズムのスケーラビリティが高いです。しかしながら、機械学習アルゴリズムは反復的な傾向があります。つまり、キャッシュを利用してディスクからの繰り返しの読み取りを防止することができます。そのため、Vertica はアルゴリズムに最も適したフォーマットでトレーニングデータをキャッシュする BLOB メモリを導入しました。BLOB メモリは、反復から反復に渡す必要がある中間データを格納するためにも使用されます。ユーザーは、リソースマネージャーを介して BLOB メモリに予約されているメモリの量を制御することができ、メモリに収まらないリソースやリソースバジェットを超えるデータは自動的にディスクに流出します。これにより、限られたリソースを持つシステムでも機械学習のクエリを実行できます。

## 最適化された並列性

機械学習アルゴリズムは、ほとんどの場合、CPU に依存しています。アルゴリズムの速度を向上させるには、計算を並列化することが重要となります。機械学習アルゴリズムは、クエリの並列性を決定するための Vertica のビルトインメカニズムに依存しており、システム内の他のクエリのリソースを過度に侵害すること

なくクエリが効率的に行われるようにします。このようにして、Vertica は、マルチユーザー環境で、機械学習クエリを含む同時実行クエリ間でリソースを賢くバランスさせることができます。

## 分散モデルストレージ

機械学習モデルは、Vertica 独自の分散ファイルシステム（DFS）でクラスタ全体に複製されたファイルとして保存されます。モデルは、高速アクセスのために Vertica のネイティブフォーマットで表現されています。時間が経つと、多数のモデルが蓄積される可能性があります。モデルを簡単に管理できるように、モデルは、テーブルと同じレベルのアクセス制御をサポートするネイティブ Vertica オブジェクトとして作成され、モデルをリスティング、変更、および削除するために使い慣れた SQL 構文がサポートされます。

## 最先端の分散機械学習アルゴリズム

研究者や開発者は、既存の分散型機械学習アルゴリズムを常に改善し、新しいものを開発しています。導入したすべてのアルゴリズムについて、業界の最新の進歩を調査し、実装を Vertica のインフラストラクチャに合わせました。Vertica のカラムストア、プロジェクション、データ圧縮のおかげで、COUNT、AVG、MIN / MAX などの解析関数は非常に効率的です。それらは、正規化や欠損値の補完など、多くのデータ準備関数の要素になっています。計算上高価なパーセンタイルやカウントのようないくつかの関数では、それらを高速化するために近似バージョンが開発されました。

上記の設計アプローチにより、Vertica の機械学習がスケーラブルかつ高速であることが実現しています。Vertica の機械学習をまだ試されていないとしても、簡単に始めることができます。機械学習はデフォルトで Vertica に組み込まれています。インストールや設定は不要です。Vertica 機械学習の詳細については、Vertica documentation を参照してください。

このブログやその他の Vertica 製品に関するご意見は、ContactVertica@hpe.com までお寄せください。


© 2018 Micro Focus. All rights reserved. Micro Focus and the Micro Focus logo, among others, are trademarks or registered trademarks of Micro Focus or its subsidiaries or affiliated companies in the United Kingdom, United States and other countries. All other marks are the property of their respective owners.