Machine Learning Mondays: Vertica における機械学習のためのデータ準備

原文はこちら

はじめに

機械学習（ML）は反復的なプロセスです。データを理解し、データを準備し、モデルを構築し、モデルを展開してモデルをテストすることから、あらゆる方法でデータを慎重に調べ、操作する必要があります。これは、生データをクリーニングしてモデリングの準備をする必要があるこのサイクルの始めに特に当てはまります。このブログでは、Vertica で利用可能なすべてのデータラングリングのツールについて簡単に説明します。Vertica で ML を実施しようとする人は、以下の機能を利用してデータを準備する必要があります。

さまざまなデータソースからさまざまな形式のデータを読み込む

ほとんどのデータサイエンティストは、さまざまな情報源からのすべてのデータを 1 つの場所に集め、それらを 1 つの共通フォーマットに変換することが最も難しいステップの 1 つであると考えているでしょう。ML データは通常データフレーム形式で表されるため、Vertica のようなリレーショナルデータベースのテーブル形式は ML に適しています。

長年にわたり、Vertica は、テキスト区切り形式、JSON、Regex、ORC、Avro、Parquet、さらには Shapefile など、さまざまな形式のデータを抽出するための包括的なパーサーを開発しました。たとえば、次のステートメントは圧縮された Twitter データ Vertica にロードします。

```
=> COPY twitter FROM '/server1/TWITTER/tweets1.json.bz2' BZIP parser
fjsonparser() direct;
```

Vertica は、ローカルファイルシステム、SQL 互換データソース、HDFS、Spark、S3 や ABS などのクラウドストレージなど、さまざまなストレージシステムからのデータを受け入れます。Vertica は HDFS と Spark との緊密な統合を実現しています。これらの製品と Vertica 間のデータ転送は高度に最適化されています。

次のステートメントは、HDFS のデータにマップする外部テーブル 'hadoopExample'を作成します。その後、Vertica の他のテーブルと同じようにテーブルを読み取ることができます。

```
=> CREATE EXTERNAL TABLE hadoopExample (A VARCHAR(10), B INTEGER, C
INTEGER, D INTEGER)
AS COPY SOURCE Hdfs(
url=http://hadoop01:50070/webhdfs/v1/user/hadoopUser/example/output/*',
username='hadoopUser');
```

強力な SQL クエリエンジン

Vertica にデータを格納する最も重要な点は、ユーザーが高速 SQL エンジンを利用できることです。ユーザーは、必要に応じてそのデータをスライス＆ダイスすることができます。行/列のフィルタリング、表の結合、グループ化、順序付けはスケーラビリティが高く、洗練された処理ロジックを表現するために使用できます。これらの操作は、通常、非 SQL 環境でより冗長な実装を必要とします。

このステートメントは、各地域の各従業員が累計した休暇日のリストを生成します。

```
=> SELECT s.store_region, SUM(e.vacation_days) totalVacationDays
   FROM public.employee_dimension e
   JOIN store.store_dimension s ON s.store_region=e.employee_region
   GROUP BY s.store_region ORDER BY totalVacationDays;
```

豊富なビルトイン関数

Vertica には、数学的計算、文字列操作、日付/時刻、集約、正規表現、その他多くの関数を含む、あらゆるタイプのデータ変換を実行する何百もの関数があります。データを変換したり、データに関する統計情報を取得したりするには、1 つ関数を呼び出すだけです。次のステートメントは、テーブルの列のデータ分散の良いアイデアを提供します。

```
=> SELECT COUNT(salary), AVG(salary), STDDEV(salary), MIN(salary),
    APPROXIMATE_PERCENTILE(salary using parameters percentile=.25) p25,
    APPROXIMATE_MEDIAN(salary) median,
    APPROXIMATE_PERCENTILE(salary using parameters percentile=.75) p75,
    MAX(salary) FROM employees;
```

Vertica は、時系列分析やイベントシリーズのセッション化など、高度なデータ分析を行うための機能を拡張しています。このシンプルなステートメントは、各株式相場表示機の 2 秒間隔での入札価格を線形補間します。

```
=> SELECT slice_time, TS_FIRST_VALUE(bid, 'LINEAR') bid FROM Tickstore
   TIMESERIES slice_time AS '2 seconds' OVER (PARTITION BY symbol ORDER
BY ts);
```

機械学習のための特別なデータ準備機能

ML の共通データ変換を容易にするために、Vertica は専用関数も開発しました。Vertica 8.1 には、正規化、欠損値の補完、不均衡なデータ処理、外れ値の検出、サンプリングなどの機能があります。

正規化は、データ準備の一般的なステップです。たとえば、「employees」テーブルの「salary」カラムと「years_of_experience」カラムを正規化するには、「robust_zscore」メソッドを使用してカラムを正規化し、結果を「employees_norm」ビューに保存する次の文を実行します。

```
=> SELECT NORMALIZE('employees_norm', 'employees', 'salary,
years_of_experience', 'robust_zscore');
```

モデルが正規化されたデータで訓練されている場合、予測データは予測関数に供給される前に同じ方法で正規化される必要があります。そのため、NORMALIZE_FIT、APPLY_NORMALIZE、および REVERSE_NORMALIZE のような他の関数も用意されています。NORMALIZE_FIT は、正規化されたパラメーターを生成し、モデルとして格納します。次のステートメントは、「employees_normrz」モデルの各列の中央値と MAD スコアを格納します。

```
=> SELECT NORMALIZE_FIT('employees_normrz', 'employees',
'salary,years_of_experience', 'robust_zscore');
```

APPLY_NORMALIZE は、それらを予測関数に供給する前に、予測データに同じ正規化パラメーターを適用します。

```
=> CREATE TABLE new_employees_norm AS SELECT APPLY_NORMALIZE (salary,
years_of_experience USING PARAMETERS model_name = 'employees_normrz')
FROM new_employees;
```

REVERSE_NORMALIZE を呼び出して、正規化されたデータを元の値に戻すことができます。たとえば、Kmeans でクラスター化する前に各列を正規化する必要があります。したがって、得られたクラスタの中心は正規化された値で表されます。中心をよりよく解釈するには、REVERSE_NORMALIZE を呼び出して中心値を元の縮尺に戻すことができます。

```
=> CREATE TABLE customer_segments AS SELECT REVERSE_NORMALIZE (revenue,
profit, number_of_employees  USING PARAMETERS model_name =
'customers_norm') FROM customers_segments_norm;
```

最後に

このブログが、データサイエンティストの方々がデータを機械学習のために利用できるようにするために、Vertica のすべてのツールを経験いただく手助けとなることを願っています。このブログや Vertica 製品に関するその他のご意見は、ContactVertica@hpe.com までお寄せください。