SQL 개발자를 위한 데이터베이스내 R 분석(자습서)

원문:https://docs.microsoft.com/ko-kr/sql/advanced-analytics/tutorials/sqldev-in-database-r-for-sql-developers?view=sql-server-2016

검토 및 편집: 김정선(jskim@sqlroad.com), Microsoft Data Platform MVP

이 자습서의 목표는 SQL 프로그래머에게 SQL Server에서 머신 러닝 솔루션을 직접 구축하는 경험을 제공하는 것입니다. 이 자습서에서는 저장 프로시저내에 R 코드를 포함시켜서 응용 프로그램 또는 BI 솔루션에 R을 통합하는 방법을 학습합니다.

> 참고
> Python 에서도 동일한 솔루션을 사용할 수 있습니다. SQL Server 2017 이 필요합니다. Python 개발자를 위한 database 내 분석을 참조하세요.

개요

전체 처리 과정을 구축하는 프로세스는 일반적으로 데이터 수집, 정리, 데이터 탐색, 특성 엔지니어링, 모델 교육 및 튜닝, 최종적으로 실제 환경에서의 모델 배포로 구성됩니다. 실제 코드의 개발 및 테스트는 전용 개발 환경을 사용하여 수행하는 것이 가장 좋습니다. R의 경우 RStudio 또는 R Tools for Visual Studio를 의미 할 수 있습니다.

그러나 솔루션을 만든 후에는 익숙한 Management Studio 환경에서 Transact-SQL 저장 프로시저를 사용하여 SQL Server에 쉽게 배포 할 수 있습니다.

이 자습서에서는 솔루션에 필요한 모든 R 코드를 받았다고 가정하고, SQL Server를 사용하여 솔루션을 작성하고 배포하는 데 중점을 둡니다.

- 1 단원: 예제 데이터 다운로드

  샘플 데이터 세트 및 샘플 SQL 스크립트 파일을 로컬 컴퓨터로 다운로드합니다.

- 2 단원: SQL Server PowerShell 을 사용하여 데이터 가져오기

  SQL Server 2017 인스턴스에서 데이터베이스와 테이블을 만들고 샘플 데이터를 테이블에 로드하는 PowerShell 스크립트를 실행합니다.

- 3 단원: 데이터를 탐색하고 시각화하기.

  Transact-SQL 저장 프로시저에서 R 패키지 및 함수를 호출하여 기본 데이터 탐색 및 시각화를 수행합니다.

- 4 단원: T-SQL 을 사용하여 데이터 특성 만들기

  사용자 정의 SQL 함수를 사용하여 새 데이터 특성을 만듭니다.

- 5 단원: T-SQL 을 사용하여 R 모델 학습 및 저장하기

  저장 프로시저에서 R 을 사용하여 머신 러닝 모델을 작성합니다. SQL Server 테이블에는 모델을 저장합니다.

- 6 단원: 모델 운용

  모델을 데이터베이스에 저장한 후 저장 프로시저를 사용하여 Transact-SQL 에서 예측을 위해 모델을 호출합니다.

시나리오

이 자습서는 뉴욕 시 택시 여행을 기반으로 하는 잘 알려진 공용 데이터 세트를 사용합니다. 샘플 코드를 더 빨리 실행하기 위해 데이터의 대표적인 1 % 샘플링을 만들었습니다. 이 데이터를 사용하여 시간, 거리 및 승차 위치와 같은 열을 기반으로 특정 여행이 팁을 얻는 지 여부를 예측하는 이진 분류 모델을 작성합니다.

요구 사항

이 자습서는 데이터베이스 및 테이블 만들기, 테이블로 데이터 가져오기, SQL 쿼리 작성 등 기본적인 데이터베이스 작업을 잘 알고 있는 사용자를 위한 것입니다. 모든 R 코드가 제공되므로 R 개발 환경은 필요하지 않습니다. 숙련된 SQL 프로그래머는 SQL Server Management Studio에서 Transact-SQL을 사용하고 제공된 PowerShell 스크립트를 실행하여 이 예제를 완성할 수 있어야 합니다. 그러나 이 자습서를 시작하기 전에 다음 준비 작업을 완료해야 합니다.

- R 서비스를 사용하여 SQL Server 2016 의 인스턴스에 연결하거나 Machine Learning Services 및 R 을 사용하여 SQL Server 2017 에 연결합니다.

- 이 자습서에서 사용하는 로그인에는 데이터베이스 및 기타 오브젝트를 작성하고, 데이터를 업로드하고, 데이터를 선택하고, 저장 프로시저를 실행할 수 있는 권한이 있어야합니다.

참고

SQL Server Management Studio 를 사용하여 R 코드를 작성하거나 테스트**하지 않는 것**이 좋습니다. 저장 프로시저에 포함 된 코드에 문제가 있는 경우 저장 프로시저에서 반환되는 정보는 일반적으로 오류의 원인을 이해하기에 부적합합니다.
디버깅을 위해 R Tools for Visual Studio 또는 RStudio 와 같은 도구를 사용하는 것이 좋습니다. 이 자습서에서 제공되는 R 스크립트는 기존의 R 도구를 사용하여 이미 개발되고 디버깅 되었습니다.

다음 단원

1 단원: 예제 데이터 다운로드

1 단원: 예제 데이터 다운로드

이 문서는 SQL Server에서 R을 사용하는 방법에 대한 SQL 개발자를 위한 자습서의 일부입니다.

이 단계에서는 이 자습서에서 사용되는 예제 데이터 세트와 Transact-SQL 스크립트 파일을 다운로드합니다. 데이터 파일과 스크립트 파일은 모두 GitHub에서 공유되지만 PowerShell 스크립트는 데이터 및 스크립트 파일을 원하는 로컬 디렉토리에 다운로드합니다.

데이터와 스크립트 다운로드

1. Windows PowerShell 명령 콘솔을 엽니다.

   대상 디렉터리를 만들거나 지정된 대상에 파일을 쓰기 위해 관리자 권한이 필요한 경우 **관리자 권한으로 실행** 옵션을 사용합니다.

2. *DestDir* 매개변수 값을 로컬 디렉터리로 변경하여 다음 PowerShell 명령을 실행합니다. 여기서 사용한 기본값은 **TempRSQL** 입니다.

   역주) 아래 팁 부분을 먼저 참조해서 수행하는 것이 보다 수월합니다.

```
$source = 'https://raw.githubusercontent.com/Azure/Azure-MachineLearning-DataScience/master/Misc/RSQL/Download_Scripts_SQL_Walkthrough.ps1'
$ps1_dest = "$pwd₩Download_Scripts_SQL_Walkthrough.ps1"
$wc = New-Object System.Net.WebClient
$wc.DownloadFile($source, $ps1_dest)
.₩Download_Scripts_SQL_Walkthrough.ps1 –DestDir 'C:₩tempRSQL'
```

   DestDir 에 지정한 폴더가 존재하지 않을 경우 PowerShell 스크립트를 통해 생성됩니다.

> 팁

오류가 발생할 경우 Bypass 인수를 사용하고 변경 범위를 현재 세션으로 지정하여 이 연습에 대해서만 PowerShell 스크립트 실행 정책을 일시적으로 **무제한**으로 설정할 수 있습니다.

역주) 아래 명령 실행 시 정책 변경에 대한 최종 확인 메시가 출력되면, 'Y'를 입력합니다.

복사

```
Set-ExecutionPolicy Bypass -Scope Process
```

이 명령을 실행해도 구성이 변경되지는 않습니다.

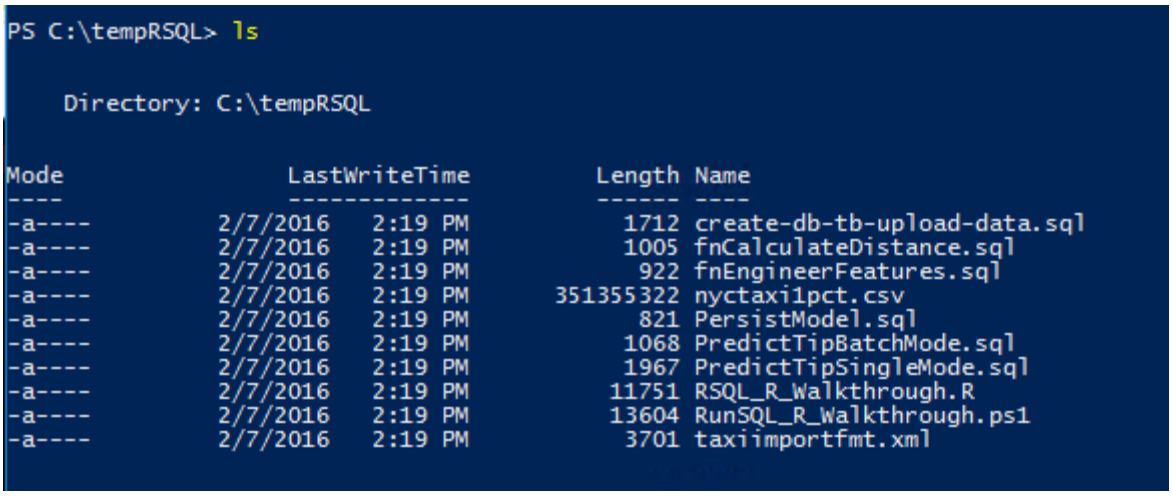
인터넷 연결에 따라 다운로드하는 데 시간이 걸릴 수도 있습니다.

3. 모든 파일이 다운로드되면 PowerShell 스크립트가 *DestDir*에 지정된 폴더에서 열립니다. PowerShell 명령 프롬프트에서 다음 명령을 실행하고 다운로드된 파일을 검토합니다.

복사

```
ls
```

Results:



다음 단원

2 단원: SQL Server PowerShell을 사용하여 데이터 가져오기

이전 단원

SQL 개발자를 위해 데이터베이스 내에서 R 분석

2 단원: SQL Server PowerShell 을 사용하여 데이터 가져오기

이 문서는 SQL Server에서 R을 사용하는 방법에 대한 SQL 개발자를 위한 자습서의 일부입니다.

이 단계에서는 다운로드한 스크립트 중 하나를 실행하여 연습에 필요한 데이터베이스 개체를 만듭니다. 또한 사용할 대부분의 저장 프로시저를 만들고 지정한 데이터베이스의 테이블에 샘플 데이터를 업로드합니다.

SQL 개체를 만드는 스크립트를 실행합니다.

다운로드 한 파일 중에서 연습을 위한 환경을 준비하기 위해 실행할 수 있는 PowerShell 스크립트가 있어야 합니다. 스크립트가 수행하는 작업은 다음과 같습니다.

- 아직 설치되지 않은 경우 SQL Native Client 및 SQL 명령 줄 유틸리티 설치. 이러한 유틸리티는 **bcp** 를 사용하여 데이터베이스에 데이터를 대량로드하는 데 필요합니다
- SQL Server 인스턴스에 데이터베이스 및 테이블 만들기, 테이블에 데이터 대량 삽입
- SQL 함수 및 저장 프로시저 만들기

스크립트 실행

1. 관리자 권한으로 PowerShell 명령 프롬프트를 열고 다음 명령을 실행합니다.

```
ps 복사

.₩RunSQL_SQL_Walkthrough.ps1
```

다음 정보를 입력하라는 메시지가 표시됩니다.

- SQL Server 2017 이 설치된 R Services(In-Database) 인스턴스의 이름 또는 주소.
- 인스턴스의 계정에 대한 사용자 이름 및 암호. 계정에는 데이터베이스를 만들고 테이블과 저장 프로시저를 만들고 테이블에 데이터를 업로드할 수 있는 권한이 있어야합니다. 사용자 이름과 암호를 입력하지 않으면 Windows ID 가 SQL Server 에 로그인하는 데 사용됩니다.
- 방금 다운로드한 샘플 데이터 파일의 경로 및 파일 이름. 예를 들어

  C:₩tempRSQL₩nyctaxi1pct.csv

2. 이 단계의 일부로, 스크립트 입력으로 제공하는 데이터베이스 이름과 사용자 이름으로 자리 표시자를 바꾸기 위해 모든 Transact-SQL 스크립트도 수정됩니다.

   스크립트를 통해 생성된 저장 프로시저 및 함수를 검토합니다.

SQL 스크립트 파일 이름	**함수**
create-db-tb-upload-data.sql	데이터베이스와 두 개의 테이블을 만듭니다. *nyctaxi_sample*: NYC 택시 데이터 세트의 1% 샘플인 학습용 데이터를 저장하는 테이블입니다. 저장소와 쿼리 성능 향상을 위해 클러스터형 columnstore 인덱스가 테이블에 추가됩니다. *nyc_taxi_models*: 학습된 모델을 이진 형식으로 저장하는데 사용되는 테이블입니다.
fnCalculateDistance.sql	승하차 위치 간의 직접 거리를 계산하는 SQL 스칼라 반환 함

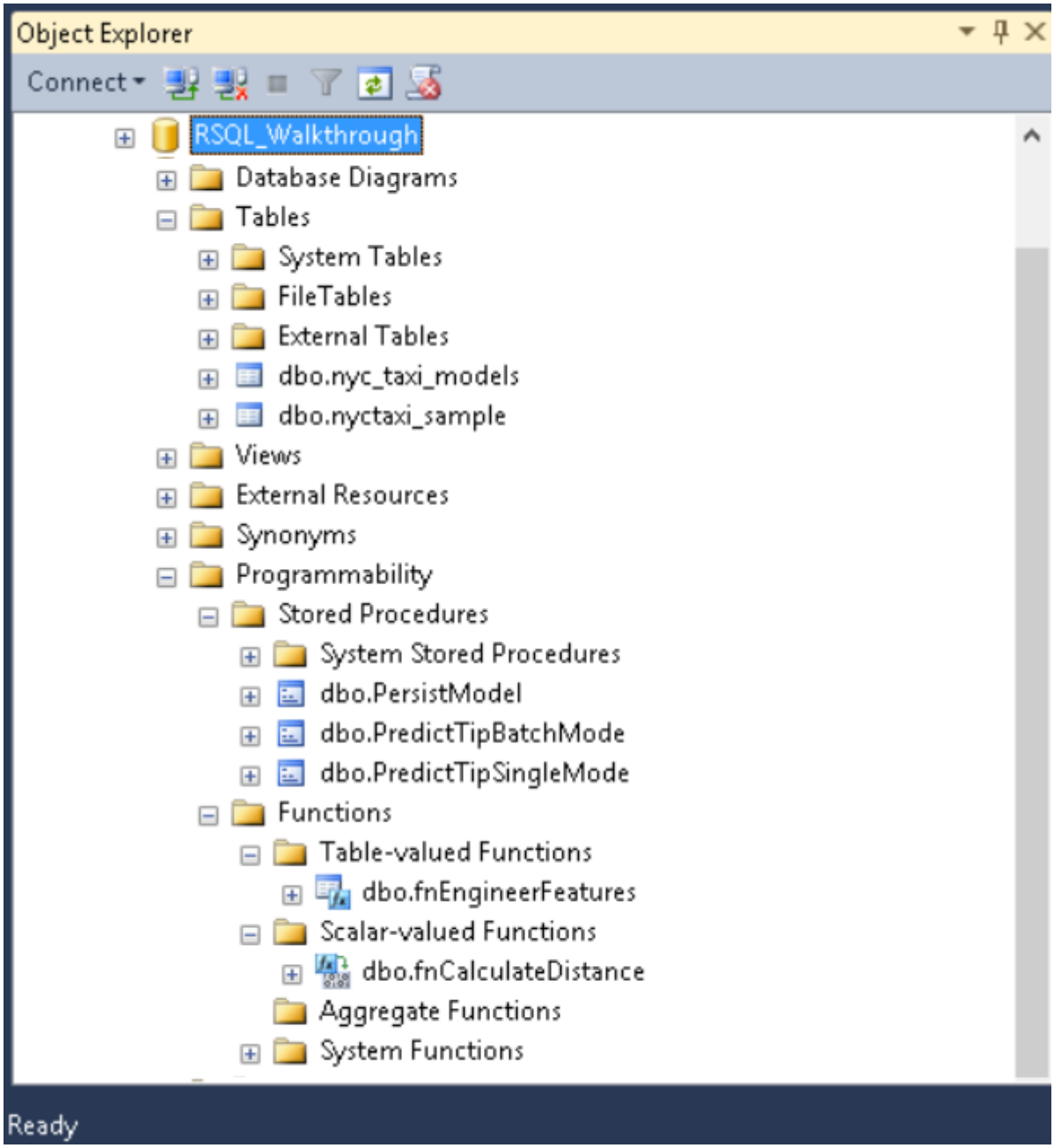
SQL 스크립트 파일 이름	함수
	수를 만듭니다.
fnEngineerFeatures.sql	분류 모델 학습을 위해 특성(feature)를 만드는 SQL 테이블 반환 함수를 만듭니다.
PersistModel.sql	모델을 저장하기 위해 호출할 수 있는 저장 프로시저를 만듭니다. 저장 프로시저는 varbinary 데이터 형식으로 직렬화된 모델을 사용하고 지정된 테이블에 씁니다.
PredictTipBatchMode.sql	학습된 모델을 호출하여 모델을 사용한 예측을 만드는 저장 프로시저를 만듭니다. 저장 프로시저는 쿼리를 입력 매개변수로 사용하고 입력된 행에 대한 점수를 포함하는 숫자 값 열을 반환합니다.
PredictTipSingleMode.sql	학습된 모델을 호출하여 모델을 사용한 예측을 만드는 저장 프로시저를 만듭니다. 이 저장 프로시저는 새 관찰을 개별 특성 값이 인라인 매개변수로 전달되는 입력으로 사용하고 새 관찰의 결과를 예측하는 값을 반환합니다.

이 연습의 뒷부분에서는 몇 개의 저장 프로시저를 추가로 만듭니다.

SQL 스크립트 파일 이름	함수
PlotHistogram.sql	데이터 탐색을 위한 저장 프로시저를 만듭니다. 이 저장 프

SQL 스크립트 파일 이름	함수
	로시저는 R 함수를 호출하여 변수의 히스토그램을 그린 다음 그 그림을 이진 개체로 반환합니다.
PlotInOutputFiles.sql	데이터 탐색을 위한 저장 프로시저를 만듭니다. 이 저장 프로시저는 R 함수를 사용하여 그래픽을 만든 다음 출력을 로컬 PDF 파일로 저장합니다.
TrainTipPredictionModel.sql	R 패키지를 호출하여 로지스틱 회귀 모델을 학습하는 저장 프로시저를 만듭니다. 모델은 tipped 열의 값을 예측하며 임의로 선택된 70%의 데이터를 사용하여 학습됩니다. 저장 프로시저의 출력은 nyc_taxi_models 테이블에 저장된 학습된 모델입니다.

3. SQL Server 및 지정한 로그인으로 SQL Server Management Studio 인스턴스에 로그인하여 생성된 데이터베이스, 테이블, 함수 및 저장 프로시저를 볼 수 있는지 확인합니다.



참고

데이터베이스 개체가 이미 있는 경우에는 다시 만들 수 없습니다.

테이블이 이미 있는 경우 데이터가 추가되며 덮어쓰지 않습니다. 따라서 스크립트를 실행하기 전에 기존 개체를 모두 삭제해야 합니다.

다음 단원

3 단원: 데이터를 탐색하고 시각화하기

이전 단원

1 단원: 예제 데이터 다운로드

3 단원: 데이터를 탐색하고 시각화하기

이 문서는 SQL 개발자를 위한 SQL Server에서 R을 사용하는 방법에 관한 자습서의 일부입니다.

이 단원에서는 예제 데이터를 검토하고 R 함수를 사용하여 일부 플롯을 생성합니다. 이러한 R 함수 R Services(In-Database)에 이미 포함됩니다. Transact-SQL에서 R 함수를 호출할 수 있습니다.

데이터 검토

일반적으로 데이터 과학 솔루션 개발에는 데이터 탐색 및 데이터 시각화가 많이 포함됩니다. 먼저 잠시 시간을 내서 샘플 데이터를 검토해 보겠습니다.

원본 데이터 세트에서는 택시 식별자 및 여행 기록이 별도의 파일로 제공되었습니다. 그러나 샘플 데이터를 쉽게 사용하기위해 두 개의 원래 데이터 세트가 *medallion*, *hack_license* 및 *pickup_datetime* 열에 조인되었습니다. 레코드도 원래 레코드 수의 1%만 가져오도록 샘플링되었습니다. 다운 샘플링된 결과 데이터 세트에는 1,703,957개의 행과 23개 열이 있습니다.

택시 식별자

- *medallion* 열은 택시의 고유 ID 번호를 나타냅니다.
- *heck_license* 열은 택시 운전 면허 번호를 포함 합니다(익명으로 처리).

여정 및 요금 레코드

- 각 여정 레코드에는 승하차 위치 및 시간과 여정 거리가 포함됩니다.
- 각 요금 레코드에는 지불 유형, 총 지불 금액, 팁 금액 등의 지불 정보가 포함됩니다.
- 마지막 세 열은 다양한 머신 러닝 작업에 사용될 수 있습니다. *tip_amount* 열은

  연속적인 숫자 값을 포함하며 회귀 분석의 **레이블** 열로 사용할 수 있습니다. *tipped* 열은

예 / 아니오 값만 있고 이진 분류에 사용됩니다. *tip_class* 열은 여러 개의 **클래스 레이블**을 가지므로 다중 클래스 분류 작업의 레이블로 사용할 수 있습니다.

이 연습에서는 이진 분류 작업만 보여줍니다. 다른 두 가지 머신 러닝 작업, 회귀 및 다중 클래스 분류에 대한 모델을 작성하는 것도 좋습니다.

- 레이블 열에 사용 되는 값은 모두 아래 비즈니스 규칙을 사용한 *tip_amount* 열에 기반합니다:

파생 열 이름	규칙
tipped	If tip_amount > 0, tipped = 1, otherwise tipped = 0
tip_class	Class 0: tip_amount = $0
	Class 1: tip_amount > $0 and tip_amount <= $5
	Class 2: tip_amount > $5 and tip_amount <= $10
	Class 3: tip_amount > $10 and tip_amount <= $20
	Class 4: tip_amount > $20

T-SQL 에서 R 을 사용하여 플롯 만들기

시각화는 데이터와 이상값의 분포를 파악하는 데 강력한 도구이므로 R에서는 데이터 시각화를 위한 여러 패키지를 제공합니다. R의 표준 오픈 소스 배포에는 히스토그램, 산점

도, 상자 그림 및 기타 데이터 탐색 그래프를 만들기 위한 많은 함수도 포함되어 있습니다.

일반적으로 R은 그래픽 출력에 R 장치를 사용하여 이미지를 만듭니다. 이 장치의 출력을 캡처한 다음 응용 프로그램에서 렌더링하기 위해 이미지를 **varbinary** 데이터 형식으로 저장하거나, 이미지를 지원 파일 형식(.JPG, .PDF 등) 중 하나로 저장할 수 있습니다.

이 섹션에서는 저장 프로시저를 사용하여 각 형식의 출력으로 작업하는 방법을 알아봅니다. 전체 프로세스는 다음과 같습니다.

- varbinary 데이터로 R 플롯을 생성하는 저장 프로시저 만들기
- 플롯을 생성하고 이미지 파일에 저장
- 저장 프로시저를 사용하여 이진 플롯 데이터를 JPG 또는 PDF 파일로 변환

PlotHistogram 저장 프로시저 만들기

1. 플롯을 만들려면 R Services(In-Database)에서 제공하는 향상된 R 함수 중의 하나인 rxHistogram 을 사용하여 Transact-SQL 쿼리의 데이터를 기반으로 히스토그램을 그립니다. R 함수를 쉽게 호출할 수 있도록 저장 프로시저 *PlotHistogram* 내에 래핑합니다.

   SQL Server Management Studio 에서, 새 **쿼리** 창을 엽니다.

2. 자습서 데이터가 포함된 데이터베이스에 아래 문을 사용하여 프로시저를 만듭니다.

```
SQL                                                                    복사

CREATE PROCEDURE [dbo].[PlotHistogram]
AS
BEGIN
  SET NOCOUNT ON;
  DECLARE @query nvarchar(max) =
```

```
  N'SELECT tipped FROM nyctaxi_sample'
  EXECUTE sp_execute_external_script @language = N'R',
                                     @script = N'
   image_file = tempfile();
   jpeg(filename = image_file);
   #Plot histogram
   rxHistogram(~tipped, data=InputDataSet, col=''lightgreen'',
   title = ''Tip Histogram'', xlab =''Tipped or not'', ylab =''Counts'');
   dev.off();
   OutputDataSet <- data.frame(data=readBin(file(image_file, "rb"), what=raw(), n=1e6));
   ',
   @input_data_1 = @query
   WITH RESULT SETS ((plot varbinary(max)));
END
GO
```

필요한 경우 올바른 테이블 이름을 사용하도록 코드를 수정해야 합니다.

- @query 변수는 스크립트 입력 변수 @input_data_1 에 대한 인수로 R 스크립트에 전달되는 쿼리 텍스트('SELECT tipped FROM nyctaxi_sample')를 정의합니다.
- R 스크립트는 아주 간단합니다. 이미지를 저장할 R 변수(image_file)를 정의한 다음 rxHistogram 함수를 호출하여 그림을 생성합니다.
- R 장치는 **off** 로 설정됩니다.

R 에서 높은 수준의 그리기 명령을 실행하면 R 에서 *장치*라는 그래픽 창이 열립니다. 창의 크기 및 색과 기타 측면을 변경하거나, 파일에 쓰거나 혹은 기타 다른 방법으로 출력을 처리하는 경우 장치를 끌 수 있습니다.

- R 그래픽 개체는 출력을 위해 R data.frame 으로 직렬화됩니다.

그래픽 데이터를 생성하고 파일에 저장

저장 프로시저는 이미지를 varbinary 데이터의 스트립트로 반환합니다. 이 데이터는 분명히 직접 볼 수 없습니다. 그러나 **bcp** 유틸리티를 사용하여 varbinary 데이터를 가져오고 클라이언트 컴퓨터에 이미지 파일로 저장할 수 있습니다.

1. Management Studio 에서 다음 문을 실행합니다.

```
SQL                                                                    복사
```

```
EXEC [dbo].[PlotHistogram]
```

결과

plot 0xFFD8FFE000104A4649...

2. PowerShell 명령 프롬프트를 열고 인스턴스 이름, 데이터베이스 이름, 사용자 이름 및 자격 증명을 인수로 제공하여 다음 명령을 실행합니다.

```
복사
```

```
bcp "exec PlotHistogram" queryout "plot.jpg" -S <SQL Server instance name> -d <database name>   -U <user name> -P <password>
```

참고

Bcp 의 명령 스위치는 대/소문자를 구분하지 않습니다.

3. 연결에 성공하면 그래픽 파일 형식에 대한 자세한 정보를 입력하라는 메시지가 표시됩니다. 아래 변경 사항 외에는 각 프롬프트에서 Enter 키를 눌러 기본값을 적용합니다.

- **prefix-length of field plot** 에 대해 0 을 입력합니다.

- 나중에 다시 사용하기 위해 출력 매개변수를 저장할 경우 **Y** 를 입력합니다.

```
복사
```

```
Enter the file storage type of field plot [varbinary(max)]:
Enter prefix-length of field plot [8]: 0
Enter length of field plot [0]:
Enter field terminator [none]:

Do you want to save this format information in a file? [Y/n]
Host filename [bcp.fmt]:
```

결과

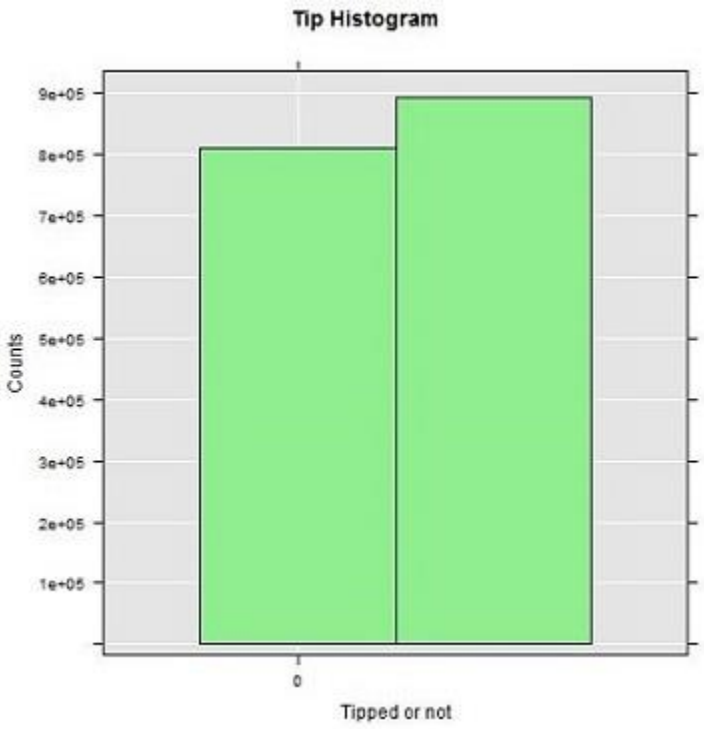
```
복사
```

```
Starting copy...
1 rows copied.
Network packet size (bytes): 4096
Clock Time (ms.) Total     : 3922   Average : (0.25 rows per sec.)
```

팁

서식 정보를 파일(bcp.fmt)에 저장하는 경우 **bcp** 유틸리티는 나중에 그래픽 파일 형식 옵션을 묻지 않고 유사한 명령에 적용할 수 있는 서식 정의를 생성합니다. 서식 파일을 사용하려면 명령줄의 끝, password 인수 뒤에 -f bcp.fmt 를 추가합니다.

4. PowerShell 명령을 실행한 곳과 동일한 디렉터리에 출력 파일이 만들어집니다. 그림을 보려면 plot.jpg 파일을 열기만 하면 됩니다.



그리기 데이터를 볼 수 있는 파일로 내보내기

R 플롯을 이진 데이터 유형으로 출력하는 것은 응용 프로그램에서 사용하기에 편리할 수 있지만 데이터 탐색 단계에서 렌더링된 플롯을 필요로 하는 데이터 과학자에게는 그다지 유용하지 않습니다. 일반적으로 데이터 과학자는 여러 관점에서 데이터에 대한 통찰력을 얻기 위해 여러 데이터 시각화를 생성합니다.

사용자용 그래프를 생성하려면 .JPG, .PDF 및 .PNG와 같은 일반적인 형식으로 R 출력을 만드는 저장 프로시저를 사용할 수 있습니다. 저장 프로시저가 그래픽을 만든 후에는 파일을 열어 플롯을 시각화하면 됩니다.

1. 히스토그램, 산점도 및 기타 R 그래픽을 JPG 및 PDF 형식으로 작성하는 방법을 보여주는 새로운 저장 프로시저 *PlotInOutputFiles*를 만듭니다.

   SQL Server Management Studio에서 새 **쿼리** 창을 열고 다음 Transact-SQL 문을 붙여 넣습니다.

SQL 복사

```
CREATE PROCEDURE [dbo].[PlotInOutputFiles]
AS
BEGIN
  SET NOCOUNT ON;
  DECLARE @query nvarchar(max) =
  N'SELECT cast(tipped as int) as tipped, tip_amount, fare_amount FROM
[dbo].[nyctaxi_sample]'
  EXECUTE sp_execute_external_script @language = N'R',
  @script = N'
   # Set output directory for files and check for existing files with same names
    mainDir <- ''C:₩₩temp₩₩plots''
    dir.create(mainDir, recursive = TRUE, showWarnings = FALSE)
    setwd(mainDir);
    print("Creating output plot files:", quote=FALSE)

    # Open a jpeg file and output histogram of tipped variable in that file.
    dest_filename = tempfile(pattern = ''rHistogram_Tipped_'', tmpdir = mainDir)
    dest_filename = paste(dest_filename, ''.jpg'',sep="")
    print(dest_filename, quote=FALSE);
    jpeg(filename=dest_filename);
    hist(InputDataSet$tipped, col = ''lightgreen'', xlab=''Tipped'',
        ylab = ''Counts'', main = ''Histogram, Tipped'');
     dev.off();

    # Open a pdf file and output histograms of tip amount and fare amount.
    # Outputs two plots in one row
    dest_filename = tempfile(pattern = ''rHistograms_Tip_and_Fare_Amount_'', tmpdir =
mainDir)
    dest_filename = paste(dest_filename, ''.pdf'',sep="")
    print(dest_filename, quote=FALSE);
    pdf(file=dest_filename, height=4, width=7);
    par(mfrow=c(1,2));
    hist(InputDataSet$tip_amount, col = ''lightgreen'',
```

```
        xlab="Tip amount ($)",
        ylab = "Counts",
        main = "Histogram, Tip amount", xlim = c(0,40), 100);
    hist(InputDataSet$fare_amount, col = "lightgreen",
        xlab="Fare amount ($)",
        ylab = "Counts",
        main = "Histogram,
        Fare amount",
        xlim = c(0,100), 100);
  dev.off();

    # Open a pdf file and output an xyplot of tip amount vs. fare amount using lattice;
    # Only 10,000 sampled observations are plotted here, otherwise file is large.
    dest_filename = tempfile(pattern = "rXYPlots_Tip_vs_Fare_Amount_", tmpdir =
mainDir)
    dest_filename = paste(dest_filename, ".pdf",sep="")
    print(dest_filename, quote=FALSE);
    pdf(file=dest_filename, height=4, width=4);
    plot(tip_amount ~ fare_amount,
        data = InputDataSet[sample(nrow(InputDataSet), 10000), ],
        ylim = c(0,50),
        xlim = c(0,150),
        cex=.5,
        pch=19,
        col="darkgreen",
        main = "Tip amount by Fare amount",
        xlab="Fare Amount ($)",
        ylab = "Tip Amount ($)");
    dev.off();',
@input_data_1 = @query
END
```

- 저장 프로시저 내의 SELECT 쿼리 출력은 기본 R 데이터 프레임인 InputDataSet 에 저장됩니다. 그런 다음 다양한 R 그리기 함수를 호출하여 실제 그래픽 파일을 생성할 수 있습니다.

포함된 R 스크립트의 대부분은 plot 또는 hist 와 같은 그래픽 함수의 옵션을 나타냅니다.

- 모든 파일은 로컬 폴더 *C:₩temp₩Plots₩에 저장됩니다.* 대상 폴더는 R 스크립트에 저장 프로시저의 일부로 제공되는 인수에 의해 정의됩니다. mainDir 변수 값을 변경하여 대상 폴더를 변경할 수 있습니다.

2. 저장 프로시저를 만드는 문을 실행합니다.

```
SQL                                                                    복사
```

```
EXEC PlotInOutputFiles
```

결과

```
                                                                       복사
```

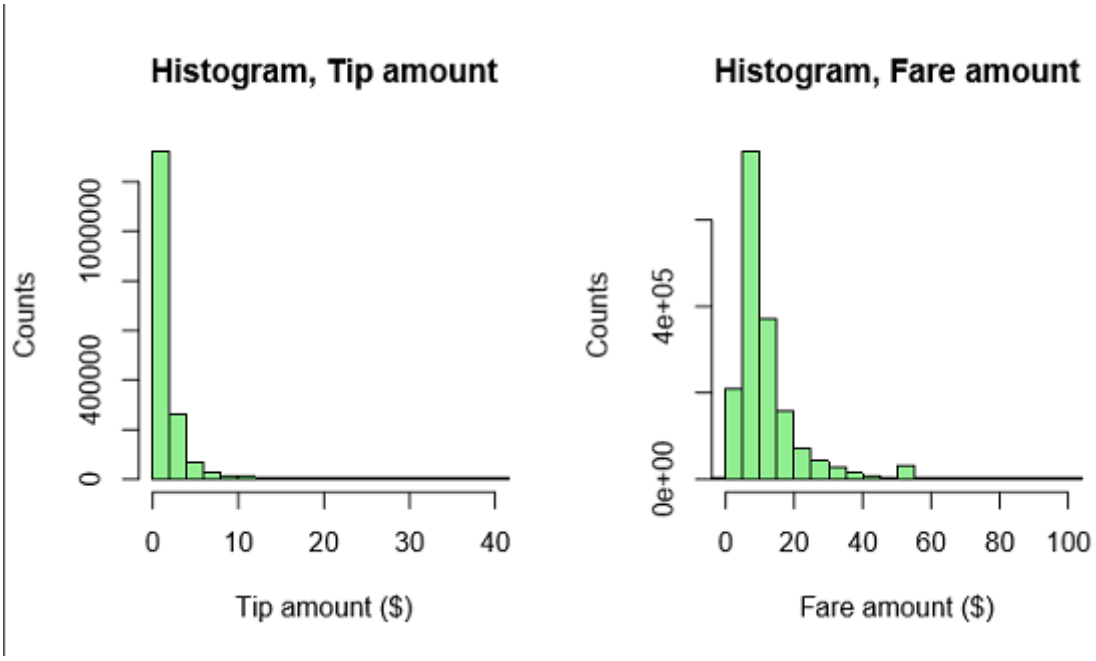
```
STDOUT message(s) from external script:
[1] Creating output plot files:[1] C:₩temp₩plots₩rHistogram_Tipped_18887f6265d4.jpg[1]

C:₩temp₩plots₩rHistograms_Tip_and_Fare_Amount_1888441e542c.pdf[1]

C:₩temp₩plots₩rXYPlots_Tip_vs_Fare_Amount_18887c9d517b.pdf
```

파일 이름에 숫자는 기존 파일에 쓸 때 오류가 발생하지 않도록 임의로 생성됩니다.

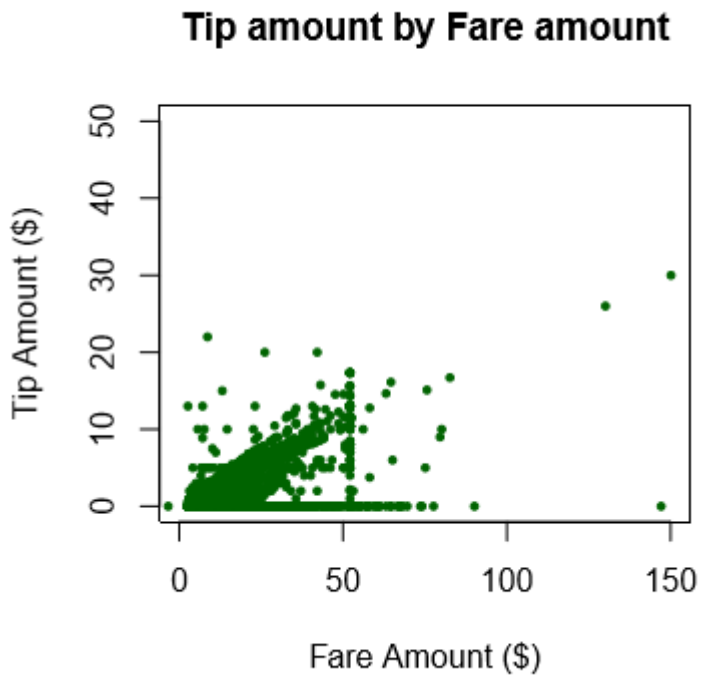
3. 플롯을 보려면 대상 폴더를 열고 저장 프로시저 내 R 코드에 의해서 생성된 파일을 검토합니다.

- rHistogram_Tipped.jpg 파일에는 팁이 있는 승차 수(Counts)와 팁이 없는 승차 수를 보여 줍니다. (이 히스토그램은 이전 단계에서 생성된 것과 매우 유사합니다.)

- rHistograms_Tip_and_Fare_Amount.pdf 파일에는 운임에 대해 그리려는 팁 금액의 분포를 보여줍니다.



- rXYPlots_Tip_vs_Fare_Amount.pdf 파일에는 X 축에 운임과 y 축에 팁 금액 가지는 산점도를 포함합니다.



4. 파일을 다른 폴더에 출력하려면 저장 프로시저에 포함된 R 스크립트의 mainDir 변수 값을 변경합니다. 다른 형식, 더 많은 파일 등을 출력하도록 스크립트를 수정할 수도 있습니다.

다음 단원

4 단원: T-SQL을 사용하여 데이터 특성 만들기

이전 단원

2 단원: SQL Server PowerShell을 사용하여 데이터 가져오기

4 단원: T-SQL 을 사용하여 데이터 특성 만들기

이 문서는 SQL Server에서 R을 사용하는 방법에 대한 SQL 개발자를 위한 자습서의 일부입니다.

이 단계에서는 Transact-SQL 함수를 사용하여 원시 데이터에서 특성을 만드는 방법을 알아봅니다. 그런 다음 저장 프로시저에서 해당 함수를 호출하여 특성 값이 포함된 테이블을 만듭니다.

특성(Feature) 엔지니어링에 대하여

데이터 탐색을 여러 번 수행한 후 데이터에서 몇 가지 유용한 정보를 수집했으며 특성 *엔지니어링*으로 넘어갈 준비가 되었습니다. 원시 데이터에서 의미 있는 특성을 만드는 이 과정은 분석 모델을 만드는 중요한 단계입니다.

이 데이터 세트에서 distance 값은 보고된 미터 거리를 기반으로 하며 이동한 지리적 거리 또는 실제 거리를 나타내는 것은 아닙니다. 따라서 NYC 택시 데이터 소스에서 사용 가능한 좌표를 사용하여 승차 위치와 하차 위치 사이의 직접 거리를 계산해야 합니다. 사용자 정의 Transact-SQL 함수에서 Haversine 수식을 사용하여 이 작업을 수행 할 수 있습니다.

사용자 정의 T-SQL 함수 *fnCalculateDistance*를 사용하여 Haversin 수식을 사용한 거리를 계산하고 두 번째 사용자 정의 함수 *fnEngineerFeatures*를 사용해서 모든 특성을 포함하는 테이블을 생성합니다.

전체 프로세스는 다음과 같습니다.

- 계산을 수행하는 T-SQL 함수 만들기
- 특성 데이터를 생성하는 함수 호출하기
- 특성 데이터를 테이블에 저장하기

FnCalculateDistance 를 사용한 이동 거리 계산

이 자습서 준비의 일부로 *fnCalculateDistance* 함수를 다운로드하고 SQL Server에 등록해야 합니다. 잠시 시간을 내어 코드를 검토하십시오.

1. Management Studio에서 **프로그래밍 기능**, **함수**, **스칼라 반환 함수**를 차례로 확장합니다.
2. *fnCalculateDistance*를 마우스 오른쪽 단추로 클릭하고 **수정**을 선택하여 새 쿼리 창에서 Transact-SQL 스크립트를 엽니다.

SQL 복사

```
CREATE FUNCTION [dbo].[fnCalculateDistance] (@Lat1 float, @Long1 float, @Lat2 float,
@Long2 float)
-- User-defined function that calculates the direct distance between two geographical
coordinates.
RETURNS float
AS
BEGIN
  DECLARE @distance decimal(28, 10)
  -- Convert to radians
  SET @Lat1 = @Lat1 / 57.2958
  SET @Long1 = @Long1 / 57.2958
  SET @Lat2 = @Lat2 / 57.2958
  SET @Long2 = @Long2 / 57.2958
  -- Calculate distance
  SET @distance = (SIN(@Lat1) * SIN(@Lat2)) + (COS(@Lat1) * COS(@Lat2) *
COS(@Long2 - @Long1))
  --Convert to miles
  IF @distance <> 0
  BEGIN
    SET @distance = 3958.75 * ATAN(SQRT(1 - POWER(@distance, 2)) / @distance);
  END
  RETURN @distance
END
GO
```

- 함수는 스칼라 반환 함수이며, 미리 정의된 형식의 단일 데이터 값을 반환합니다.

- 승하차 위치에서 얻은 위도 및 경도 값을 입력으로 사용합니다. Haversine 수식은 위치를 라디안으로 변환하고 해당 값을 사용하여 두 위치 사이의 직접 거리(마일)를 계산합니다.

fnEngineerFeatures 사용한 특성 생성

모델을 학습하는데 사용할 수 있는 테이블에 계산된 값을 추가하려면 *fnEngineerFeatures*라는 다른 함수를 사용합니다. 새 함수는 이전에 생성된 T-SQL 함수 *fnCalculateDistance*를 호출하여 승차 지점과 하차 위치 사이의 직접적인 거리를 구합니다.

1. 잠깐 시간을 내어 이 연습을 준비하는 과정에서 작성해야 하는 사용자 정의 T-SQL 함수인 *fnEngineerFeatures*에 대한 코드를 검토하십시오.

SQL 복사

```
CREATE FUNCTION [dbo].[fnEngineerFeatures] (
@passenger_count int = 0,
@trip_distance float = 0,
@trip_time_in_secs int = 0,
@pickup_latitude float = 0,
@pickup_longitude float = 0,
@dropoff_latitude float = 0,
@dropoff_longitude float = 0)
RETURNS TABLE
AS
  RETURN
  (
  -- Add the SELECT statement with parameter references here
  SELECT
    @passenger_count AS passenger_count,
    @trip_distance AS trip_distance,
    @trip_time_in_secs AS trip_time_in_secs,
```

```
    [dbo].[fnCalculateDistance](@pickup_latitude, @pickup_longitude, @dropoff_latitude,
@dropoff_longitude) AS direct_distance

  )
GO
```

- 이 테이블 반환 함수는 여러 열을 입력으로 취하고 여러 특성 열을 가진 테이블을 반환합니다.
- 이 함수의 목적은 모델 작성에 사용할 새로운 특성을 만드는 것입니다.

2. 이 함수가 작동하는지 확인하려면 측정 거리가 0이지만 승차 및 하차 위치가 다른 이동의 지리적 거리를 계산하는데 사용하십시오.

SQL 복사

```
    SELECT tipped, fare_amount, passenger_count,(trip_time_in_secs/60) as TripMinutes,
    trip_distance, pickup_datetime, dropoff_datetime,
    dbo.fnCalculateDistance(pickup_latitude, pickup_longitude,  dropoff_latitude,
dropoff_longitude) AS direct_distance
    FROM nyctaxi_sample
    WHERE pickup_longitude != dropoff_longitude and pickup_latitude !=
dropoff_latitude and trip_distance = 0
    ORDER BY trip_time_in_secs DESC
```

보시다시피 측정기가 보고한 거리가 지리적 거리와 항상 일치하는 것은 아닙니다. 이것이 특성 엔지니어링이 중요한 이유입니다. 이러한 향상된 데이터 특성으로 R을 사용한 머신 러닝 모델을 학습할 수 있습니다.

다음 단원

5 단원: T-SQL을 사용한 모델 학습 및 저장

이전 단원

3 단원: 데이터 탐색 및 시각화

5 단원: T-SQL 을 사용한 모델 학습 및 저장

이 문서는 SQL Server에서 R을 사용 하는 방법에 대 한 SQL 개발자를 위한 자습서의 일부입니다.

이 단원에서는 R을 사용하여 머신 러닝 모델을 학습시키는 방법을 배웁니다. 방금 만든 데이터 특성을 사용하여 모델을 학습시킨 다음 학습된 모델을 SQL Server 테이블에 저장합니다. 이 경우 R 패키지는 이미 R Services (In-Database)와 함께 설치되어 있으므로 SQL로 모든 것을 수행할 수 있습니다.

저장 프로시저 만들기

T-SQL에서 R을 호출할 때 시스템 저장 프로시저 sp_execute_external_script를 사용합니다. 그러나 모델 재학습과 같이 자주 반복되는 프로세스의 경우 다른 저장 프로시저에서 `sp_execute_exernal_script`에 대한 호출을 캡슐화 는 것이 보다 쉽습니다.

1. 먼저 팁 예측 모델을 작성하는 R 코드를 포함하는 저장 프로시저를 만듭니다. Management Studio에서 새 **쿼리** 창을 열고 다음 문을 실행하여 *TrainTipPredictionModel* 저장 프로시저를 만듭니다. 이 저장 프로시저는 입력 데이터를 정의하고 R 패키지를 사용하여 로지스틱 회귀 모델을 만듭니다.

SQL 복사

```
CREATE PROCEDURE [dbo].[TrainTipPredictionModel]

AS
BEGIN
  DECLARE @inquery nvarchar(max) = N'
    select tipped, fare_amount, passenger_count,trip_time_in_secs,trip_distance,
    pickup_datetime, dropoff_datetime,
    dbo.fnCalculateDistance(pickup_latitude, pickup_longitude,  dropoff_latitude,
dropoff_longitude) as direct_distance
    from nyctaxi_sample
    tablesample (70 percent) repeatable (98052)
'
```

```
    -- Insert the trained model into a database table
    INSERT INTO nyc_taxi_models
    EXEC sp_execute_external_script @language = N'R',
                                    @script = N'

## Create model
logitObj <- rxLogit(tipped ~ passenger_count + trip_distance + trip_time_in_secs +
direct_distance, data = InputDataSet)
summary(logitObj)

## Serialize model and put it in data frame
trained_model <- data.frame(model=as.raw(serialize(logitObj, NULL)));
',
    @input_data_1 = @inquery,
    @output_data_1_name = N'trained_model'
    ;

END
GO
```

- 모델을 테스트하기 위해 일부 데이터를 남겨야 하므로, 택시 데이터 테이블에서 70 %를 무작위로 선택합니다.
- SELECT 쿼리는 사용자 정의 스칼라 함수 *fnCalculateDistance*를 사용하여 승차 및 하차 위치 사이의 직접 거리를 계산합니다. 쿼리의 결과는 기본 R 입력 변수인 InputDataset에 저장됩니다.
- R 스크립트는 R 서비스(In-Database)에 포함된 향상된 R 기능 중 하나인 rxLogit 함수를 호출하여 로지스틱 회귀 모델을 만듭니다.

  이진 변수 *tipped*는 *레이블* 또는 결과 열로 사용되며 모델은 *passenger_count, trip_distance, trip_time_in_secs* 및 *direct_distance*와 같은 특성 열을 사용하여 적합(fitting)합니다.
- R 변수 logitObj에 저장되어있는 학습된 모델은 직렬화해서 SQL Server로 출력하기 위해 데이터 프레임에 저장됩니다. 이 출력은 데이터베이스 테이블

nyc_taxi_models에 삽입되므로 향후 예측에 사용할 수 있습니다.

2. 저장 프로시저가 존재하지 않는 경우 다음 문을 실행해서 저장 프로시저를 만듭니다.

저장 프로시저를 사용한 R 모델 생성

저장 프로시저에 이미 입력된 데이터의 정의가 포함되므로 입력 쿼리를 제공할 필요가 없습니다.

1. R 모델을 생성하려면 다른 매개변수없이 저장 프로시저를 호출합니다.

SQL 복사

```
EXEC TrainTipPredictionModel
```

2. Management Studio **메시지** 창에서 아래 메시지와 같이 R의 stdout 스트림으로 파이프될 메시지를 봅니다.

   "STDOUT message(s) from external script: Rows Read: 1193025, Total Rows Processed: 1193025, Total Chunk Time: 0.093 seconds "

   모델 생성의 일부로 생성된 변수 및 테스트 메트릭을 표시하는 개별 함수 rxLogit에 대한 메시지를 볼 수도 있습니다.

3. 문이 완료되면 *nyc_taxi_models* 테이블을 엽니다. 데이터 처리 및 모델 적합에는 다소 시간이 걸릴 수 있습니다.

   model 열에 직렬화된 모델을 포함하는 새 행이 추가된 것을 확인할 수 있습니다.

복사

```
model
------
0x580A00000002000302020....
```

다음 단계에서는 학습된 모델을 사용하여 예측을 만듭니다.

다음 단원

6 단원: 모델 운용

이전 단원

4 단원: T-SQL을 사용한 데이터 특성 만들기

6 단원: R 모델 운용

이 문서는 SQL Server에서 R을 사용하는 방법에 대한 SQL 개발자를 위한 자습서의 일부입니다.

이 단계에서는 저장 프로시저를 사용하여 모델을 *운용(operationalize)*하는 방법을 학습합니다. 이 저장 프로시저는 다른 응용 프로그램에서 직접 호출하여 새 관측에 대한 예측을 할 수 있습니다. 이 연습에서는 저장 프로시저에서 R 모델을 사용하여 채점하는 두 가지 방법을 보여줍니다.

- **일괄 처리 채점 모드**: 저장 프로시저에 대한 입력으로 SELECT 조회를 사용하십시오. 저장 프로시저는 입력된 사례(case)에 해당하는 관측 표를 반환합니다.
- **개별 채점 모드**: 일련의 개별 매개변수 값을 입력으로 전달합니다. 저장 프로시저는 단일 행이나 값을 반환합니다.

먼저 채점의 일반적인 작동 방식을 살펴보겠습니다.

기본 채점

PredictTip 저장 프로시저는 예측을 수행하는 기본 구문을 보여줍니다.

SQL 복사

```
CREATE PROCEDURE [dbo].[PredictTip] @inquery nvarchar(max)
AS
BEGIN

DECLARE @lmodel2 varbinary(max) = (SELECT TOP 1 model FROM nyc_taxi_models);
EXEC sp_execute_external_script @language = N'R',
  @script = N'
    mod <- unserialize(as.raw(model));
    print(summary(mod))
    OutputDataSet<-rxPredict(modelObject = mod, data = InputDataSet, outData =
NULL, predVarNames = "Score", type = "response", writeModelVars = FALSE, overwrite =
TRUE);
```

```
    str(OutputDataSet)
    print(OutputDataSet)
    ',
  @input_data_1 = @inquery,
  @params = N'@model varbinary(max)',
  @model = @lmodel2
  WITH RESULT SETS ((Score float));
END
GO
```

- SELECT 문은 데이터베이스에서 직렬화된 모델을 가져와서, R을 사용한 추가 처리를 위해 R 변수 mod에 그 모델을 저장합니다.
- 새로운 채점용 사례는 저장 프로시저의 첫 번째 매개변수인 @inquery에 지정된 Transact-SQL 쿼리에서 가져옵니다. 쿼리에서 읽은 데이터는 기본 데이터 프레임인 InputDataSet에 저장됩니다. 이 데이터 프레임은 점수를 생성하는 R의 rxPredict 함수에 전달됩니다.

  ```
  OutputDataSet<-rxPredict(modelObject = mod, data = InputDataSet, outData = NULL, predVarNames = "Score", type = "response", writeModelVars = FALSE, overwrite = TRUE);
  ```

  data.frame은 단일 행을 포함 할 수 있으므로 일괄 처리 또는 단일 점수 지정에 동일한 코드를 사용할 수 있습니다.
- rxPredict 함수에 의해 반환되는 값은 운전자가 임의의 팁을 얻을 확률을 나타내는 float입니다..

일괄 처리 채점

이제 일괄 처리 채점의 작동 방식을 살펴보겠습니다.

1. 작은 입력 데이터 세트를 사용하여 작업을 시작하겠습니다. 이 쿼리는 예측에 필요한 승객 수와 기타 특성들로 "상위 10개"의 여정 목록을 만듭니다.

```
SQL 복사
```

```
SELECT TOP 10 a.passenger_count AS passenger_count, a.trip_time_in_secs AS trip_time_in_secs, a.trip_distance AS trip_distance, a.dropoff_datetime AS dropoff_datetime, dbo.fnCalculateDistance(pickup_latitude, pickup_longitude, dropoff_latitude,dropoff_longitude) AS direct_distance

FROM (SELECT medallion, hack_license, pickup_datetime, passenger_count,trip_time_in_secs,trip_distance, dropoff_datetime, pickup_latitude, pickup_longitude, dropoff_latitude, dropoff_longitude FROM nyctaxi_sample)a

LEFT OUTER JOIN

(SELECT medallion, hack_license, pickup_datetime FROM nyctaxi_sample TABLESAMPLE (70 percent) REPEATABLE (98052) )b

ON a.medallion=b.medallion AND a.hack_license=b.hack_license

AND a.pickup_datetime=b.pickup_datetime

WHERE b.medallion IS NULL
```

예제 결과

복사

passenger_count	trip_time_in_secs	trip_distance	dropoff_datetime	direct_distance
1	283	0.7	2013-03-27 14:54:50.000	0.5427964547
1	289	0.7	2013-02-24 12:55:29.000	0.3797099614
1	214	0.7	2013-06-26 13:28:10.000	0.6970098661

이 쿼리는 다운로드의 일부로 제공되는 *PredictTipBatchMode* 저장 프로시저에 대한 입력으로 사용할 수 있습니다.

2. Management Studio에서 *PredictTipBatchMode* 저장 프로시저의 코드를 검토합니다.

SQL 복사

```
/****** Object:  StoredProcedure [dbo].[PredictTipBatchMode]  ******/
CREATE PROCEDURE [dbo].[PredictTipBatchMode] @inquery nvarchar(max)
```

```
AS
BEGIN
DECLARE @lmodel2 varbinary(max) = (SELECT TOP 1 model FROM nyc_taxi_models);
EXEC sp_execute_external_script
  @language = N'R',
  @script = N'
    mod <- unserialize(as.raw(model));
    print(summary(mod))
    OutputDataSet<-rxPredict(modelObject = mod, data = InputDataSet, outData =
NULL, predVarNames = "Score", type = "response", writeModelVars = FALSE, overwrite =
TRUE);
    str(OutputDataSet)
    print(OutputDataSet)
  ',
  @input_data_1 = @inquery,
  @params = N'@model varbinary(max)',
  @model = @lmodel2
  WITH RESULT SETS ((Score float));
END
```

3. 변수에 쿼리 텍스트를 입력하고 저장 프로시저에 매개변수로 전달합니다.

SQL 복사

```
-- Define the input data
DECLARE @query_string nvarchar(max)
SET @query_string='SELECT TOP 10 a.passenger_count as passenger_count,
a.trip_time_in_secs AS trip_time_in_secs, a.trip_distance AS trip_distance,
a.dropoff_datetime AS dropoff_datetime, dbo.fnCalculateDistance(pickup_latitude,
pickup_longitude, dropoff_latitude,dropoff_longitude) AS direct_distance FROM  (SELECT
medallion, hack_license, pickup_datetime, passenger_count,trip_time_in_secs,trip_distance,
dropoff_datetime, pickup_latitude, pickup_longitude, dropoff_latitude, dropoff_longitude
FROM nyctaxi_sample  )a  LEFT OUTER JOIN (SELECT medallion, hack_license,
pickup_datetime FROM nyctaxi_sample TABLESAMPLE (70 percent) REPEATABLE (98052))b
ON a.medallion=b.medallion AND a.hack_license=b.hack_license AND
a.pickup_datetime=b.pickup_datetime WHERE b.medallion is null'
```

```
-- Call the stored procedure for scoring and pass the input data
EXEC [dbo].[PredictTip] @inquery = @query_string;
```

4. 저장 프로시저는 상위 10 개 여정 각각에 대한 예측을 나타내는 일련의 값을 반환합니다. 그러나, 상위 여정은 운전자가 팁을 얻지 못할 가능성이 상대적으로 짧은 여행 거리를 가진 단일 승객 여행이기도 합니다.

팁

"yes-tip"및 "no-tip"결과만 반환하는 대신 예측에 대한 확률 점수를 반환하고 *Score* 열 값에 WHERE 절을 적용하여 점수를 0.5 또는 0.7과 같은 임계 값을 사용한 "팁을 줄 가능성이 높음"또는 "주지 않을 가능성이 있음"으로 분류 할 수 있습니다. 이 단계는 저장 프로시저에 포함되지 않지만 쉽게 구현할 수 있습니다.

단일 행 채점

때로는 응용 프로그램에서 개별 값을 전달하고 해당 값을 기반으로 단일 결과를 얻고 싶을 수도 있습니다. 예를 들어 저장 프로시저를 호출하고 사용자가 입력하거나 선택한 입력을 제공하도록 Excel 워크 시트, 웹 응용 프로그램 또는 Reporting Services 보고서를 설정할 수 있습니다.

이 섹션에서는 저장 프로시저를 사용하여 단일 예측을 만드는 방법을 설명합니다.

1. 다운로드의 일부로 포함된 *PredictTipSingleMode* 저장 프로시저 의 코드를 검토합니다.

SQL 복사

```
CREATE PROCEDURE [dbo].[PredictTipSingleMode] @passenger_count int = 0,
@trip_distance float = 0, @trip_time_in_secs int = 0, @pickup_latitude float = 0,
@pickup_longitude float = 0, @dropoff_latitude float = 0, @dropoff_longitude float = 0
AS
BEGIN
```

```
DECLARE @inquery nvarchar(max) = N'SELECT * FROM
[dbo].[fnEngineerFeatures](@passenger_count, @trip_distance, @trip_time_in_secs,
@pickup_latitude, @pickup_longitude, @dropoff_latitude, @dropoff_longitude)';
DECLARE @lmodel2 varbinary(max) = (SELECT TOP 1 model FROM nyc_taxi_models);
EXEC sp_execute_external_script
  @language = N'R',
  @script = N'
    mod <- unserialize(as.raw(model));
    print(summary(mod));
    OutputDataSet<-rxPredict(modelObject = mod, data = InputDataSet, outData =
NULL, predVarNames = "Score", type = "response", writeModelVars = FALSE, overwrite =
TRUE);
    str(OutputDataSet);
    print(OutputDataSet);
    ',
  @input_data_1 = @inquery,
  @params = N'@model varbinary(max),@passenger_count int,@trip_distance
float,@trip_time_in_secs int ,   @pickup_latitude float ,@pickup_longitude
float ,@dropoff_latitude float ,@dropoff_longitude float', @model = @lmodel2,
@passenger_count =@passenger_count, @trip_distance=@trip_distance,
@trip_time_in_secs=@trip_time_in_secs, @pickup_latitude=@pickup_latitude,
@pickup_longitude=@pickup_longitude, @dropoff_latitude=@dropoff_latitude,
@dropoff_longitude=@dropoff_longitude
  WITH RESULT SETS ((Score float));
END
```

- 이 저장 프로시저는 승객 수, 이동 거리 등의 여러 단일 값을 입력으로 사용합니다.

  외부 응용 프로그램에서 저장 프로시저를 호출하는 경우 데이터가 R 모델의 요구 사항 일치 하는지 확인합니다. 여기에는 입력 데이터를 R 데이터 형식으로 변환 또는 변환 데이터 유형 및 데이터 길이의 유효성을 검증 할 수 있는지 여부가 포함됩니다.

- 저장 프로시저는 저장된 R 모델을 기반으로 점수를 만듭니다.

2. 수동으로 값을 제공하여 시험해 보세요.

새 **쿼리** 창을 열고 저장 프로시저를 호출하여 각 매개변수의 값을 제공합니다. 매개변수

는 모델에서 사용하는 특성 열을 나타내며 필수 항목입니다.

복사

```
EXEC [dbo].[PredictTipSingleMode] @passenger_count = 0,
@trip_distance = 2.5,
@trip_time_in_secs = 631,
@pickup_latitude = 40.763958,
@pickup_longitude = -73.973373,
@dropoff_latitude =  40.782139,
@dropoff_longitude = 73.977303
```

또는 저장 프로시저에 매개변수에서 지원되는 짧은 형식을 사용하십시오:

SQL 복사

```
EXEC [dbo].[PredictTipSingleMode] 1, 2.5, 631, 40.763958,-73.973373, 40.782139,-73.977303
```

3. 결과는 상위 10 번의 여정에서 팁을 얻을 확률이 낮다는 것을 의미합니다. 왜냐하면 모두가 비교적 짧은 거리에 걸쳐 단일 승객의 여정이기 때문입니다.

결론

이것으로 자습서를 마칩니다. 저장 프로시저에 R 코드를 포함하는 방법을 배웠음으로 이러한 연습을 확장하여 자신의 모델을 빌드할 수 있습니다. Transact-SQL과의 통합으로 예측을 위해 R 모델을 배포하고 엔터프라이즈 데이터 워크플로의 일부로 모델 재교육을 통합하는 것이 훨씬 쉬워졌습니다.

이전 단원

5 단원: T-SQL 을 사용한 R 모델 학습 및 저장