Predictive analytics using IBM SPSS Modeler in DB2 for z/OS

Jane Man (janeman@us.ibm.com)
Senior Software Engineer
IBM

Lei Tian (tianlei@cn.ibm.com)
Staff Software Engineer
IBM

Liang Wang (wanglcdl@cn.ibm.com)
Senior Software Engineer
IBM

29 May 2015

This tutorial focuses on predictive analytics using IBM SPSS® Modeler and data stored in DB2® for z/OS®. We illustrate how to use the SPSS Modeler workbench to create predictive models with in-database mining, SQL push-back, and UDF scoring. We walk through the steps for integrating real-time scoring for DB2 for z/OS into an OLTP application. We also show what needs to be done in the DB2 server and the information an application developer needs to know to create an enterprise solution for in-database transactional scoring and batch scoring.

Introduction

DB2 for z/OS is commonly used to store reliable transaction data critical for real-time scoring. IBM SPSS Modeler provides a user-friendly GUI to build, test, and deploy predictive models using the data stored in DB2 for z/OS. It uses the technique SQL push-back, which avoids extracting data from the database to SPSS Modeler for scoring. It provides two ways to do scoring inside a database: with or without using the Server Scoring Adapter (Modeler UDFs).

The example in this tutorial depends on data in a sample table called DRUG1n, which you can create in your DB2 for z/OS system. The table contains information about patients given treatments for certain illnesses and patient characteristics, such as cholesterol and blood pressure.

The description of how to create and populate the table appears in the Appendix. Alternatively, you can use your own data.

In the following sections, we will cover the following:
• Setup and configuration
• Building a simple model
• Scoring inside DB2 for z/OS via Modeler UDF SQL
• Scoring inside DB2 for z/OS via pure SQL push-back
• Publishing the model into DB2 for z/OS
• Creating an SQL statement to do in-database real-time scoring

Setup and configuration

To enable SPSS support in DB2 10 for z/OS, you need:

• IBM DB2 for z/OS 10 NFM with APARs to enable built-in functions PACK and UNPACK:
  • PM55928 for PACK/UNPACK preconditioning
  • PM56631 for enabling
  • PM74654 Scalability and performance improvements
• IBM Modeler With Scoring Adapter for zEnterprise V16 (PID 5655-AA8), which includes FMID HHUMG10: IBM SPSS Modeler Server Scoring Adapter for DB2 for z/OS
• SPSS Modeler Client V16
• IBM SPSS OEM Connect64 for ODBC 6.1

The scoring adapter contains jobs to create the database, tables, and UDFs required by the scoring adapter, the job to define and setup the WLM, and the job to bind the scoring adapter packages. See Resources for details.

Building a simple model using data stored in DB2 for z/OS

The following steps illustrate how to build a simple model using SPSS Modeler (installed in Windows) and data stored in DB2 for z/OS. For the SPSS Modeler to talk to DB2 for z/OS, we need to configure an ODBC DSN.

Step 1: Configure ODBC DSN

a. Click Start > Control Panel.
b. In the Control Panel, double-click Administrative Tools > Data Source (ODBC).
c. In the ODBC Data Source Administrator dialog box, click Add to add a new data source.
d. From the list of drivers, find IBM SPSS OEM 6.1 DB2 Wire Protocol and double-click it as shown below.
Figure 1. Selecting the SPSS DB2 Wire Protocol driver

![Create New Data Source dialog box](image)

e. Enter the data source name, IP address, TCP port, location name, and the collection of the DB2 for z/OS you intend to connect to, then click **Test Connect**. It pulls up the setup dialog box as shown below. Ensure that the following details are entered: DB2 10 for z/OS (as data source name), zserveros.demos.ibm.com with 5447 as the TCP port, EOSDB208 as the location name, and DSNT as the collection.
f. Enter the DB2 user name and the corresponding password as **Password** in their respective fields, then click **OK**.
Figure 3. Test database connection

![Logon to DB2 Wire Protocol](image)

You can now see "Connection established" if the test succeeds. Click OK to close all dialogs.

**Step 2: Build a simple model using data from DB2 for z/OS**

1. Start with launching SPSS Modeler Workbench. The default Windows location: **Start > All Programs > IBM SPSS Modeler**.
2. In Modeler Workbench, drag the Database Source node from the **Favorites** node palette to the Diagram canvas, as shown below.
3. Double-click **Database Source node** icon in the Diagram Canvas to open the node properties dialog. Now in the data source drop-down, select **Add new database connection**, as shown below.

**Figure 5. Adding a database connection**

4. In the Database Connections dialog, select **DB2 10 for z/OS** (this needs to match the data source name you created in step 1(g) above), enter the DB2 user name and password and click **Connect**. The new connection will be set up in the Connections table. Click **OK** to close this dialog.
5. In the database dialog, enter DRUG1n as the table name and click **Preview** to see the table content, as shown in Figure 8. (Note: We assume table DRUG1n is already created and populated inside DB2 for z/OS). You can connect to any table you desire, but the following steps in this section are specifically for DRUG1n table. For how to load/insert data, please refer to the Appendix.

**Figure 7. Selecting a table for analysis**

![Figure 7. Selecting a table for analysis]
6. As shown in Figure 8, the DRUG1n table contains information about the characteristics of patients (age, gender, blood pressure, cholesterol, sodium, potassium), and what medications each individual is taking. We want to find out what characteristic (e.g., cholesterol level, blood pressure) is important in determining the medication he is taking. Click OK in the table content preview dialog to close it.

7. In the Node properties dialog, under the Types tab, click Clear All Values > Read Values to read data types.

8. We are now going to build a classification model, which needs to mark one of the input fields as the Target field (also called the predictor field). In our example, modify Role of Drug Field (Column) to be Target, as shown below. Click OK to close the node properties dialog.
9. In SPSS Modeler Workbench, select the **Modeling** tab in the node palette panel.

10. Drag the **CHAID** node from the **Modeling** node palette to Diagram Canvas. Each node icon in **Modeling** node palette represents a modeling algorithm. In our example, CHAID algorithm is one **RULE INDUCTION** algorithm, which derives a decision tree or a set of rules that try to describe distinct segments within the data in relation to a target field.

11. In the Diagram Canvas, right-click **DRUG1n node**, select **Connect** in its context menu, then click the **CHAID** node icon to set up a connection link as shown below.

**Figure 11. Connecting the table to an algorithm**

12. So far, we have constructed a simple stream containing one database source node and one modeling node. We could use this stream to build our CHAID model. Right-click **CHAID** node and select **Run** in its context menu, as shown in Figure 13. We will see a gold diamond-
shaped icon generated, as shown in Figure 13; this is known as the CHAID model or CHAID nugget.

**Figure 12. Running a stream**

**Figure 13. Creating a model**

13. Double-click the golden diamond (CHAID model) and you will see the output below.

**Figure 14. CHAID model output**
The output in the above image shows that K (potassium) is important in determining the medication a person is taking in the data sample in the database.

**Scoring inside DB2 for z/OS via SPSS Modeler UDF (Server Scoring Adapter)**

We will use a pre-existing stream file (see Download) to do scoring. In this stream file, a renewal model is used to predict how likely a policy holder will renew his policy.

1. Open RenewalModel_IOD.str in SPSS Modeler Workbench via File > Open Stream.
2. Click the **Analysis node** in the first branch (it should be highlighted), then click **SQL Preview** in the SPSS Modeler Workbench toolbar, as shown below.

**Figure 15. Previewing the SQL statement generated**

SQL generation is started. Regarding the analysis node as terminal node, you will notice all upstream (previous) node icons turn purple as shown below, and the generated renewed model nugget turns purple as well, which means the scoring is completed in the database.

**Figure 16. Generating the new SQL statement for scoring**

3. The **SQL Preview** button is used to see possible SQL generation for a stream execution. Sometimes we would like to know ahead how many operations can be pushed back, so we simply click **SQL Preview** to preview. The advantage of this is that the generated SQL
expressions are not executed by clicking the SQL Preview button, which saves execution effort as well as prevents execution impact.

4. To look at the generated SQL statements, click the **show stream messages** option at the bottom of SPSS Modeler Workbench.

The stream properties dialog opens, as shown below.

**Figure 17. SQL statement generated for scoring**

![SQL Statement](image)

In the above image, the SPSS Modeler Scoring UDF SQL statement is highlighted for ease of recognition. The user-defined function name is HUMSPSS.SCORE_COMPONENT. It works together with the DB2 for z/OS built-in function PACK/UNPACK for successful scoring. The UDF package needs to be installed into DB2 for z/OS before we could use it (see DB2 for z/OS Modeler UDF Installation documents for more details). SPSS Modeler will generate UDF SQL to score by default, and we call the UDF Server Scoring Adapter in the SPSS Modeler product, as shown below.

**Figure 18. Options for generating SQL**

![Options](image)

This option above exists in CHAID model nugget, and you can double-click the renewed model nugget to have a look; it resides within the **Settings** tab.

If you can't see any UDF SQL in the **Messages** tab of the Stream properties dialog, click the **Options > Logging and Status** sub-tab of the same Stream properties dialog, and make sure **Logging options** is checked, as shown below.
Figure 19. Setting up logging properties

With these settings you can control various aspects of SQL generation in the current stream. Use these settings as the default for all your streams.

Scoring inside DB2 for z/OS via SQL push-back

As mentioned, we can also do scoring without using the SPSS Modeler UDFs. As in the previous section, we will use the same first branch of the stream (RenewalModel_IOD.str) to score via pure SQL push-back.

1. Double-click the renewed model nugget, click the Settings tab, modify the Generate SQL for this model option from Default to Score by converting to native SQL with Missing Value support, as shown below, then click OK.

Figure 20. Scoring without using SPSS Modeler UDFs

2. Click the Analysis node first (it should be highlighted), then click SQL Preview in the SPSS Modeler Workbench toolbar to start SQL generation. You will notice all upstream (previous) node icons turn purple. To look at the generated SQL statements, click the stream message button at the middle bottom of SPSS Modeler Workbench.
Figure 21 shows the SQL statement without using SPSS Modeler UDFs. You will notice a longer and more complex SQL statement compared to the case of UDF SQL (Modeler Server Scoring Adapter). This is because we are generating pure SQL this time. The pure SQL push-back only uses built-in database SQL expressions and functions to score data. The model scoring algorithm itself is translated into the corresponding SQL expressions (in this renewed model case, a lot of CASE WHEN/THEN statements, as the model contains a lot of rules), which makes much more SQL expressions are generated for the model scoring algorithm. The UDF SQL case is different, which only contains one user-defined function to represent the whole model scoring algorithm.

**Figure 21. Scoring SQL statement without using SPSS Modeler UDFs**

One advantage of UDF SQL is that it supports more models than pure SQL push-back supports. Most of the models generated in SPSS Modeler Workbench support scoring via UDF SQL, while only a few of them support scoring via pure SQL push-back. Typically, pure SQL push-back is faster when a model can be described with a few SQL expressions where the model scoring algorithm logic is simple, like in the linear regression model, but UDF SQL will outperform SQL push-back when a model is more complicated.

**Publishing the model into DB2 for z/OS**

SPSS Modeler Workbench provides a way to publish an SPSS model into DB2 for z/OS and return an example SQL statement template that can be used for real-time scoring. We will use the same first branch of the stream (RenewalModel_IOD.str) to show how to publish a model to DB2 for z/OS.

1. Double-click the **RENEWED model nugget** to open the model properties dialog, select the **File** menu, then Select **Publish for Server Scoring Adapter**, as shown below.
2. In the opened dialog below, select the corresponding database connection to publish. In our case, it is SPSS01A@DB2 10 for z/OS.

**Figure 23. Selecting the target database**

3. Enter a proper publish ID string to identify the model. The publish ID is used as the model reference in application SQL statements. For illustration purposes, we will use RENEWED_MODEL as the publish ID in this tutorial.
4. Check the option **Generate Example SQL**, then click **Browse for file** to open a file chooser, and choose a new filename (e.g. “renewed”) to store the example SQL as shown in Figure 24. You can select the **Desktop** button in File chooser dialog to store the corresponding example SQL file on desktop. Click **Save** to close the File chooser dialog and click **OK** to close the Publish dialog.

5. After a few seconds, you will find a new file with the name you specified in step 4 generated, and its content is displayed below.

**Figure 25. SQL template after publishing a model**

This is the example SQL template returned for us to use in other applications. You could see the publish ID RENEWED_MODEL, which we entered previously appears in the example SQL template, and there is an internal nested SQL statement that contains $(TABLE0) in generated example SQL template. In the next section, we will see how to replace corresponding part of the example SQL template for a real-time in-database scoring.

6. To verify that we have published our model successfully, issue the **SELECT** statement, as shown in Listing 1.

**SQL statement to verify a successful model publication**

```
SELECT * FROM HUMSPSS.PUBLISHED_COMPONENTS WHERE ID = 'RENEWED_MODEL'
```

The figure below illustrates a possible output for Listing 1. A **HASHED_ID** is generated for each model published successfully.
In this section, we will use the model we published in previous section to perform in-database real-time scoring. Listing 2 is the returned example SQL template in the previous section (RENEWED.txt).

**SQL template after a successful model publication**

```sql
SELECT T0.C0 AS C0, T0.C1 AS C1, T0.C2 AS C2, T0.C3 AS C3, T0.C4 AS C4, T0.C5 AS C5, T0.C6 AS C6, T0.C7 AS C7, T0.C8 AS C8, T0.C9 AS C9, T0.C10 AS C10, UNPACK(HUMSPSS.SCORE_COMPONENT('P', 'RENEWED_MODEL', PACK(CCSID 1208, T0.C0, T0.C1, T0.C2, T0.C3, T0.C4, T0.C5, T0.C6, T0.C7, T0.C8, T0.C9, T0.C10))).* AS (C11 VARCHAR(1) CCSID UNICODE, C12 DOUBLE) FROM (SELECT T0."MONTHS_SINCE_POLICY_CHANGE" AS C0, T0."MONTHS_SINCE_POLICY_INCEPTION" AS C1, T0."SALES_CHANNEL" AS C2, T0.AGE AS C3, T0.GENDER AS C4, T0."EMPLOYMENT_STATUS" AS C5, T0.INCOME AS C6, T0."NR_OF_COMMUNICATIONS" AS C7, T0."NR_OF_OPEN_COMPLAINTS" AS C8, T0."LATEST_NOTE_ATTITUDE" AS C9, T0."CLAIM_FILED_DENIED" AS C10 FROM ${TABLE0} T0) AS T0
```

This kind of example SQL is flexible to integrate with various kinds of business database applications. In real applications, we would need to feed the second part of the above listing with real records data (usually one row or a few rows). Below are a couple of examples:

1. Using the data that already exists in DB2 for z/OS, we can replace the first part of the SQL statement in Listing 2 with that in Listing 3 to analyze a specific AGE group (e.g., AGE = 21). The complete SQL statement is shown below.

```sql
SELECT T0."MONTHS_SINCE_POLICY_CHANGE" AS C0, T0."MONTHS_SINCE_POLICY_INCEPTION" AS C1, T0."SALES_CHANNEL" AS C2, T0.AGE AS C3, T0.GENDER AS C4, T0."EMPLOYMENT_STATUS" AS C5, T0.INCOME AS C6, T0."NR_OF_COMMUNICATIONS" AS C7, T0."NR_OF_OPEN_COMPLAINTS" AS C8, T0."LATEST_NOTE_ATTITUDE" AS C9, T0."CLAIM_FILED_DENIED" AS C10 FROM ICRG_VIEW1 T0 WHERE T0."AGE" = 21
```
Figure 27. Complete SQL statement to analyze a specific age group

```sql
INSERT INTO theTable
SELECT T0."MONTHS_SINCE_POLICY_CHANGE" AS C0,T0."MONTHS_SINCE_POLICY_INCEPTION" AS C1,T0."SALES_CHANNEL" AS C2,T0.AGE AS C3,T0.GENDER AS C4,T0."EMPLOYMENT_STATUS" AS C5,T0.INCOME AS C6,T0."NR_OF_COMMUNICATIONS" AS C7,T0."NR_OF_OPEN_COMPLAINTS" AS C8,T0."LATEST_NOTE_ATTITUDE" AS C9,T0."CLAIM_FILED_DENIED" AS C10
FROM ICRG_VIEW1 T0
WHERE T0."POLICY_HOLDER_ID" = 1032121;
```

You may see the following scoring result.

Figure 28. Scoring Output (partial) for age group 21

Figure 28 shows the partial output for our analysis. There are 230 records with AGE = 21. The last two columns are the scoring output, which illustrate how likely they are going to renew their policy.

2. Similarly, using the data that already exists in DB2 for z/OS, we can replace the first part of the SQL statement in Listing 2 with that in Listing 4 to analyze a specific policy holder (e.g., POLICY_HOLDER_ID= 1032121).

```sql
SELECT T0."MONTHS_SINCE_POLICY_CHANGE" AS C0,T0."MONTHS_SINCE_POLICY_INCEPTION" AS C1,T0."SALES_CHANNEL" AS C2,T0.AGE AS C3,T0.GENDER AS C4,T0."EMPLOYMENT_STATUS" AS C5,T0.INCOME AS C6,T0."NR_OF_COMMUNICATIONS" AS C7,T0."NR_OF_OPEN_COMPLAINTS" AS C8,T0."LATEST_NOTE_ATTITUDE" AS C9,T0."CLAIM_FILED_DENIED" AS C10
FROM ICRG_VIEW1 T0
WHERE T0."POLICY_HOLDER_ID" = 1032121;
```

Figure 29. Scoring output for policy holder 1032121

Figure 29 is the scoring output for policy holder 1032121. The scoring result (C11=$R-RENEWED=1, C12=$RC-RENEWED=0.9796) means the policy holder (POLICY_HOLDER_ID= 1032121) is likely to renew the policy, with the confidence or renewal probability of 0.9796.
3. The previous two examples are based on the fact that the real data already exists in DB2 for z/OS. For data that does not exist in DB2 for z/OS, we could also use the SPSS Modeler UDF (SCORE_COMPONENT, registered in DB2 for z/OS) to do real-time scoring.

Listing 5 shows the SQL statement used to see how likely a particular person (whose data does not exist in the database: age 21, unemployed, etc.) will renew his policy.

**SQL statement for scoring records not in the database**

```
SELECT UNPACK(HUMSPSS.SCORE_COMPONENT('P', 'RENEWED_MODEL', PACK(CCSID 1208, INT(0),
    INT(4), 'Call Center', INT(21), 'M', 'Unemployed', INT(0), INT(3), INT(0),
    INT(0), INT(5))).* AS (RENEWED VARCHAR(1) CCSID UNICODE, CONFIDENCE
     DOUBLE) FROM SYSIBM.SYSDUMMY1
```

**Figure 30. Scoring output for record that does not exist in database**

Figure 30 shows the output for Listing 5. The scoring result (RENEWED=0, CONFIDENCE=0.545) means the incoming real-time data value has a high churn risk, with a churn probability of 0.545.

The first invocation of UDF real-time scoring may take more time because there is no memory cached model, but subsequent real-time scoring of the same model (identified by the publish ID) will become faster over time.

**Summary**

This tutorial discussed how to build a predictive model using IBM SPSS Modeler Workbench with data stored in DB2 for z/OS. We illustrated how to do scoring using SPSS Modeler UDF and SQL push-back. We also showed how to publish a model into DB2 for z/OS and how to create a SQL statement for in-database real-time scoring.

**Acknowledgments**

Thanks to Robin Sun and Susan Malaika for their comments and assistance with this tutorial.

**Appendix: Load/Insert data into DB2 for z/OS**

1. Start with launching Modeler Workbench. The default Windows location is `Start > All Programs > IBM SPSS Modeler`.

2. In Modeler Workbench, drag the VAR file source node from the Favorites node palette to Diagram Canvas, setting its data path, as shown in Figure 35 (the default location of DRUG1n in Modeler 16 is C:\Program Files\IBM\SPSS\Modeler\16\Demos), then click `Open` to set the data and click `OK` to close the dialog.
3. In Modeler Workbench, drag the Database Export node from the Favorites node palette to the Diagram Canvas, connecting the VAR file source node to the database export node, as shown in Figure 32.

Figure 32. Database export node

4. Double-click the Database Export node icon in the Diagram Canvas to open the node properties dialog, then in the data source drop-box, select **Existing connection DB2 10 for z/OS** item in the drop-down and set the export table name (e.g., DRUG1n), shown in Figure 37, and click **OK** to close the node dialog.
5. Right-click the **Database Export node**, and select **Run** in its context menu to upload/insert data, as shown in Figure 34.

**Figure 34. Database export node setting**
## Downloads

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Renewal model</td>
<td>RenewalModel_IOD.zip</td>
<td>45KB</td>
</tr>
</tbody>
</table>
Resources

• IBM SPSS Modeler 16 Scoring Adapter Installation
• UNPACK function in DB2 10 for z/OS
• "pureXML in DB2 9: Which way to query your XML data?" has more examples and details for XML query in DB2.
• The DB2 area on developerWorks provides resources for architects, developers, and engineers.
• Stay current with developer technical events and webcasts focused on a variety of IBM products and IT industry topics.
• Follow developerWorks on Twitter.
• Watch developerWorks demos ranging from product installation and setup demos for beginners, to advanced functionality for experienced developers.
• Get involved in the developerWorks Community. Connect with other developerWorks users while you explore developer-driven blogs, forums, groups, and wikis.
About the authors

Jane Man

Jane Man is a senior software engineer in development team in DB2 for z/OS. She has worked on various features of DB2 for z/OS. In addition to her development work, she is the enablement focal point and is involved in many enablement activities, including creating sample applications, demos, Hands on Labs, and presenting in conferences and bootcamps, etc. Before joining DB2 for z/OS, she was a developer in IBM Content Manager. Jane is a IBM Certified System Administrator for WebSphere Application Server; IBM Certified Database Administrator for DB2 Universal Database for z/OS, Linux, UNIX and Windows; IBM Certified Solution Designer for DB2 Content Manager; IBM Certified Deployment Profession for Tivoli Storage Manager; and IBM Certified Application developer for DB2 Universal Database Family.

Lei Tian

Lei Tian is a staff software engineer with the IBM SPSS Modeler development team. He has worked on many features of SPSS Modeler and focuses on database-related features now. He is the enablement focal point for corporations using SPSS Modeler and DB2. He is passionate about innovating in technology.

Liang Wang

Liang Wang is a staff software engineer in the IBM SPSS Modeler development team. He has been dedicated to IBM SPSS Modeler since 2008. During this period, he led a team focused integrating various technologies and new algorithms into IBM SPSS Modeler, such as R, Cognos, TM1, Spatio-Temporal Prediction, General Spatio Association Rule, etc.,. He won an IBM Outstanding Technical Achievement Award for implementing the first version IBM SPSS Modeler Server Scoring Adapter for DB2 for z/OS.

© Copyright IBM Corporation 2015
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)