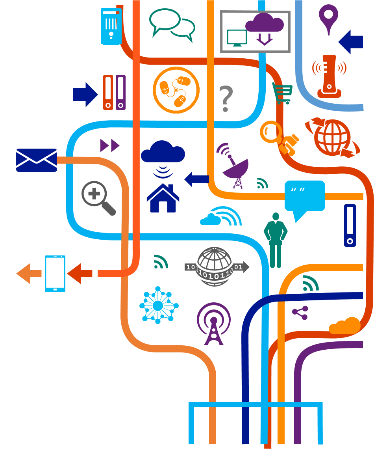
 [](http://www.netmf.com/)



IoT Kit Hands-on  
トレーニング

Step 5：分析編　（暫定版）

改定: 2015年 4月 6日

初版: 2015年 1月 8日

日本マイクロソフト株式会社

デベロッパー エクスペリエンス ＆ エバンジェリズム 統括本部

太田　寛

Twitter：@embedded\_george

Blog : http://blogs.msdn.com/hirosho

この文章に含まれる情報は、公表の日付の時点でのMicrosoft Corporationの考え方を表しています。市場の変化に応える必要があるため、Microsoftは記載されている内容を約束しているわけではありません。この文書の内容は印刷後も正しいとは保障できません。この文章は情報の提供のみを目的としています。

Microsoft、SQL Server、Visual Studio、Windows、Windows XP、Windows Server、Microsoft Azureは Microsoft Corporation の米国およびその他の国における登録商標です。

その他、記載されている会社名および製品名は、各社の商標または登録商標です。

© Copyright 2014 Microsoft Corporation. All rights reserved.

目次

[0. はじめに 4](#_Toc416034355)

[0.1 はじめに 5](#_Toc416034356)

[0.2 学習内容 6](#_Toc416034357)

[1. Excel によるデータ表示と分析 8](#_Toc416034358)

[1.1 Excel Power Queryによるデータ収集 9](#_Toc416034359)

[1.2 Excel Power Mapによるデータ可視化 14](#_Toc416034360)

[2. Stream Analyticsによる分析 17](#_Toc416034361)

[2.1 IoT Kitのコード修正 18](#_Toc416034362)

[2.2 Azure SQLによる出力用データベース作成 20](#_Toc416034363)

[2.3 Stream Analyticsサービス作成 24](#_Toc416034364)

[2.4 分析の実施 30](#_Toc416034365)

[3. Machne Learningによる分析 33](#_Toc416034366)

[3.1 Machine Learning作成 34](#_Toc416034367)

[3.2 活用アプリの作成 49](#_Toc416034368)

[4. 試してみよう 50](#_Toc416034369)

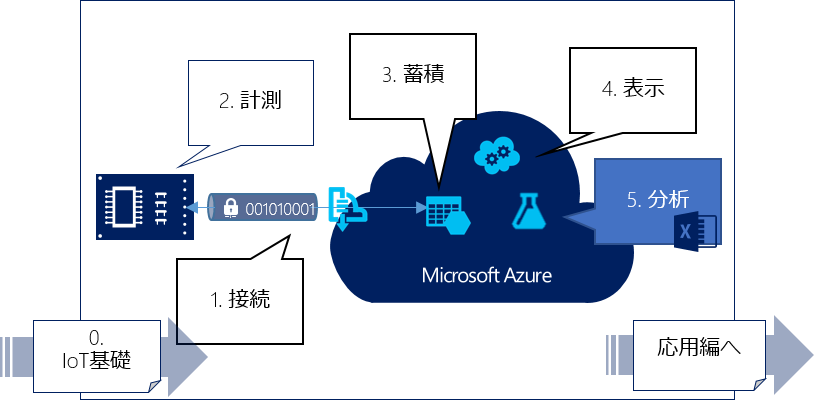
|  |
| --- |
| 1. はじめに |
|  |

## はじめに

本手順書は、一連のIoT Kit ハンズオントレーニングの五番目のステップで行う分析の実習の具体的な手順を解説します。このステップの学習を開始する前に、前のステップの学習を行ってください。

## 学習内容

このステップでは、組込み機器で計測されたセンサー値や状態を受信、あるいは、蓄積したデータの分析方法を学びます。

****

実習は以下の順番に従って行います。

1. Excel によるデータ表示と分析
   1. Excel Power Query によるデータ収集
   2. Excel Power Map による可視化
2. Stream Analyticsによるデータストリーム分析
   1. IoT Kitのコード修正
   2. SQL Azure設定
   3. Stream Analyticsサービス作成
   4. IoT Kitからセンサーデータを送信し分析を実行
3. Machine Learning による分析
   1. Machine Learningを作成する
   2. Macnine Learning を活用する
4. 試してみよう

Visual Studioの使い方や、Gadgeteerのハードウェア設計等、前のステップまでで学習した項目は、本手順書では詳しい説明を省きます。やり方が分らない場合は、前のステップに戻って再学習してください。

実習には、.NET Micro Framework / Gadgeteer 対応の小型ハードウェア（IoT Kitと記載）を使用します。本手順書で使用しているハードウェア以外でも、実習は可能です。使用するハードウェアに合わせて、適宜読み替えて実習を行ってください。

2014年12月末日現在、標準のIoT Kit ハードウェアは、GHI Electoronics 社製のFEZ Spider、及び、各種センサーを使ったFEZ版IoT Kitが販売されています。現行の手順書ではこのキットを使った手順を解説しています。GR PEACH版IoT Kitハードウェアの準備が整った段階で、改定の予定です。学習される皆様にはご迷惑をおかけしますが、ご了承ください。

|  |
| --- |
| 1. Excel によるデータ表示と分析 |
| このステップでは、IoT Kitデバイスのセンサーで計測し、Web アプリに送信しクラウドストレージに蓄積したデータを、Excelで分析する方法を実習します。   * **Excel Power Query によるデータ収集** * **Excel Power Map による可視化** |

## Excel Power Queryによるデータ収集

※この実習を行うには、

* ソフトウェアアシュアランス付 Microsoft Office 2010 Professional Plus
* Microsoft Office 2013 Professional Plus、Office 365 ProPlusまたはExcel 2013 スタンドアロン

が必要です。

AzureのクラウドストレージやSQL Server等に蓄積されたデータは、Excel Power Queryを使ってExcelへのダウンロードが可能です。Excel に一旦データをダウンロードしてしまえば、グラフ作成やPower View等を使ったビジュアライゼーションが可能になります。数字の羅列のみでは人間はそのデータの特性や傾向を把握することは困難です。IoTを活用して新たな価値を創造する過程では、データを基にした様々な試行錯誤が発生します。また、当たり前と思われているデータ傾向についても、グラフ化等による可視化で新しい発見をすることもしばしばあるでしょう。このステップの学習を通じて是非、基本スキルを獲得してください。

### Excel Power Query のインストールとセットアップ

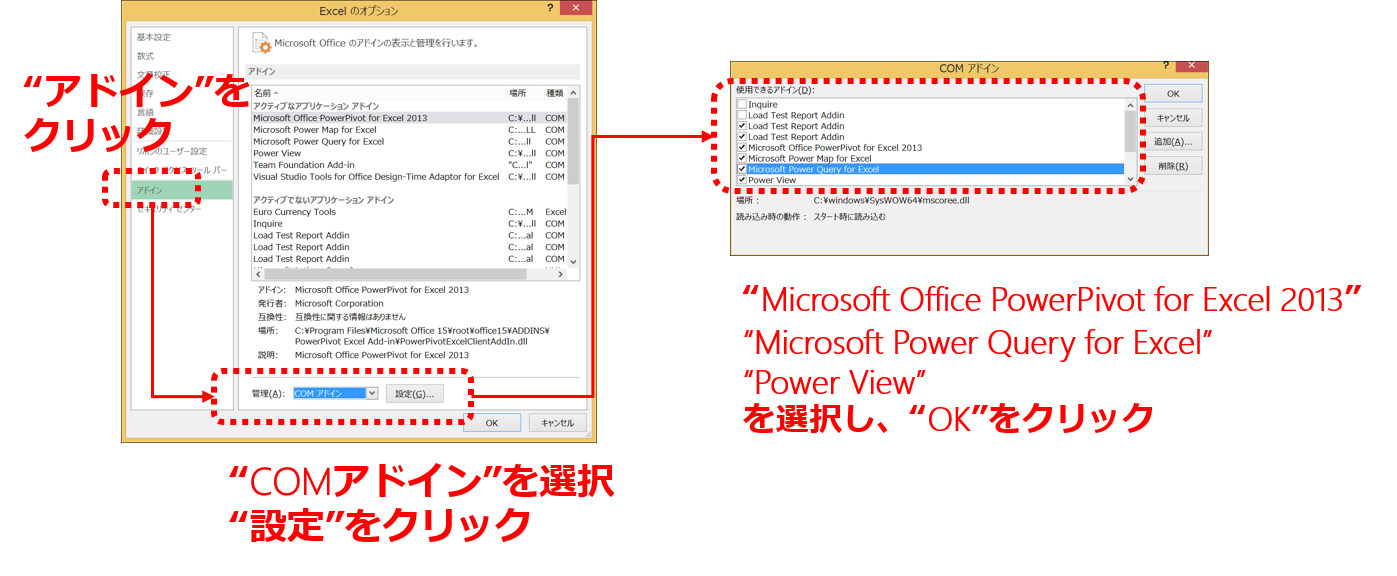
<http://www.microsoft.com/ja-jp/download/details.aspx?id=39379>

から、“Excel Power Query for Excel”をダウンロードし、インストールします。

インストールが終わったら、Excelを起動します。リボンの“ファイル”をクリックし、

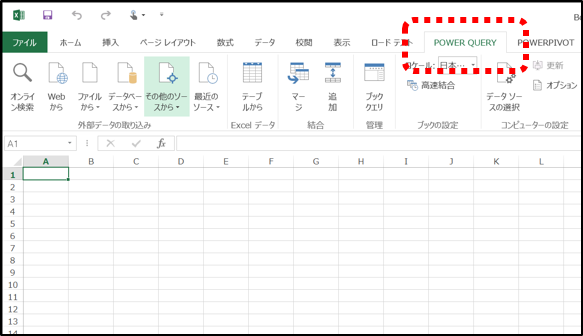


“オプション”をクリックします。



開いたダイアログで、図の通りに設定してください。これで準備完了です。

### Excel Power Query によるデータダウンロード



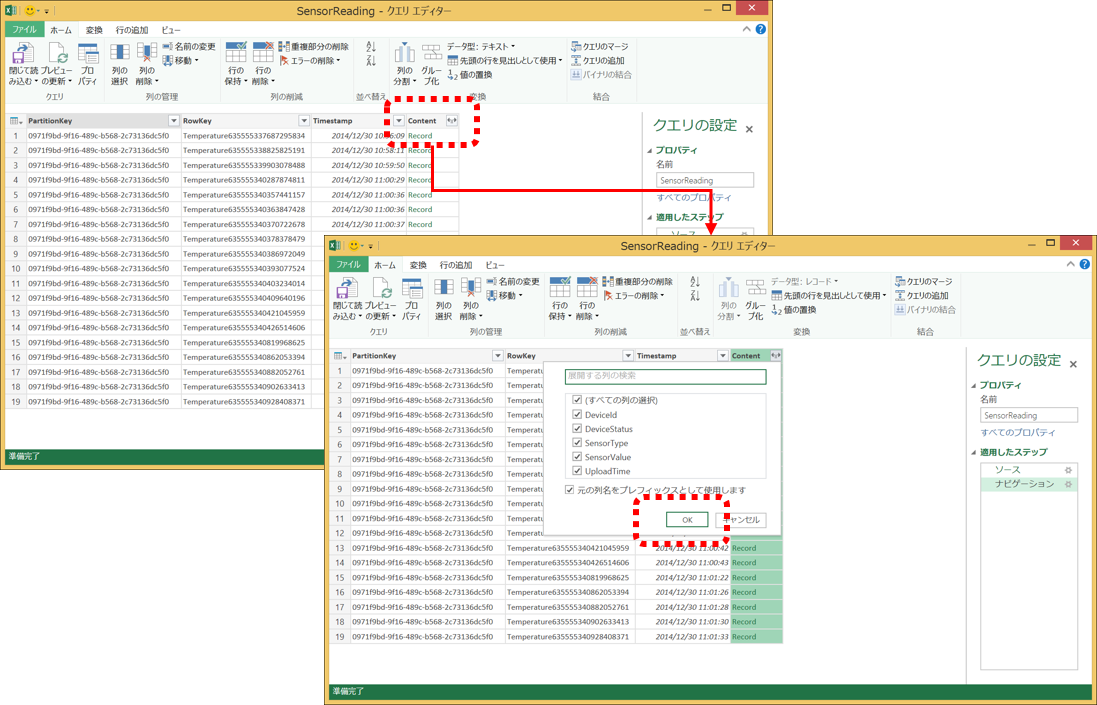
リボンの“POWER QUERY”を選択します。

“その他のソースから”をクリックすると、Azureのストレージをはじめとする各種データソースを選択できます。また、“データベースから”をクリックすれば、SQL Server（SQL Azureを含む）をはじめとする各種データベースからデータをダウンロードできます。

“その他のリソース”をクリックして、“Microsoft Azure テーブルストレージから”を選択します。

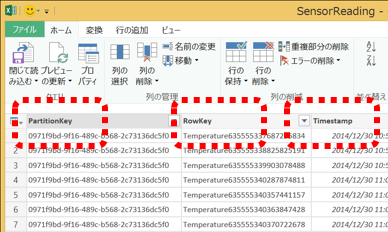


表示されたダイアログに、Step3蓄積編で作成したストレージアカウント名を入力し、“OK”をクリックします。Azureポータルで該当するストレージの“アクセスキーの管理”でセカンダリアクセスキーを“コピー＆ペーストし、”保存“をクリックします。



表示された表の、“Content”の右側の二つの矢印が付いたボタンをクリックします。すると、SensorTypeやSensorValueなどの項目が出てくるので、チェックをすべてにつけたまま、“OK”をクリックします。

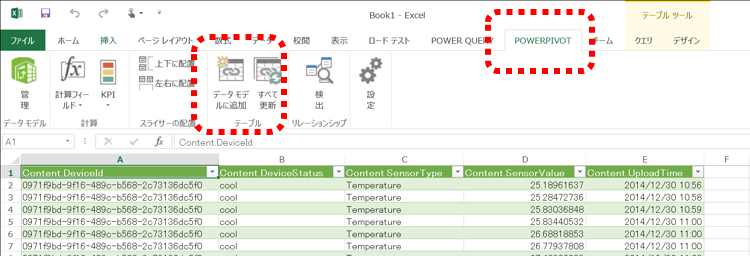
更に、



PartitionKey、RowKey、Timestampはこのトレーニングでは使用しないので、それぞれのカラムのタイトルを右クリックして“削除”を選択します。そして、“Content”の右側のボタンをクリックして表示を追加したSensorValue、UpdateTimeを右クリックして“型の変換”を選択し、それぞれ、“10進数”、“日付/時刻”に変更します。設定が終わると、



となります。左上の“閉じて読み込む”をクリックて、このシートを閉じます。



次に表示されたシートで、“POWERPIVOT”を選択し、“データモデルに追加”をクリックしておきます。

ここまでくれば、数値が読み込まれているので、後は、Excelのグラフ機能を使えば、自由自在にグラフ化できます。

Excel Power Queryの詳細は、

<https://support.office.com/ja-jp/article/Microsoft-Power-Query-for-Excel-%e3%81%ae%e6%a6%82%e8%a6%81-6e92e2f4-2079-4e1f-bad5-89f6269cd605?ui=ja-JP&rs=ja-JP&ad=JP>

をご覧ください。

### Excel Power Viewによるグラフ化

※作成中。

Excel Power Viewの詳細は、

<https://support.office.com/ja-jp/article/Office-365-%e3%81%ae-Power-View-in-Excel-%e3%81%be%e3%81%9f%e3%81%af-Power-View-in-SharePoint-Server-99127d68-d7b7-42d0-a4ee-20e2d9897472?ui=ja-JP&rs=ja-JP&ad=JP>

をご覧ください。

## Excel Power Mapによるデータ可視化

Excel Power Queryでダウンロードしたデータを地図や任意の画像と組み合わせて可視化します。IoTシナリオでは、複数の組込み機器や設備機器がクラウドに接続されます。そしてそれらの機器はそれぞれの場所に設置されています。それら機器が生み出すデータや状態を位置情報とともに可視化することにより空間情報を加味した直感的な理解を助ける可視化を行うことが可能になります。

### Excel Power Mapのインストール

<http://www.microsoft.com/en-us/download/details.aspx?id=38395>

ここからダウンロードしてインストールします。

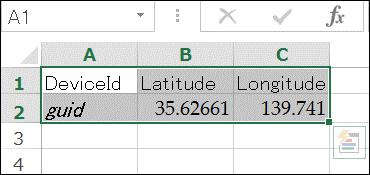
### Excel Power Mapの作成

前節でダウンロードしたセンサー計測値を基に、Power Mapを作成します。

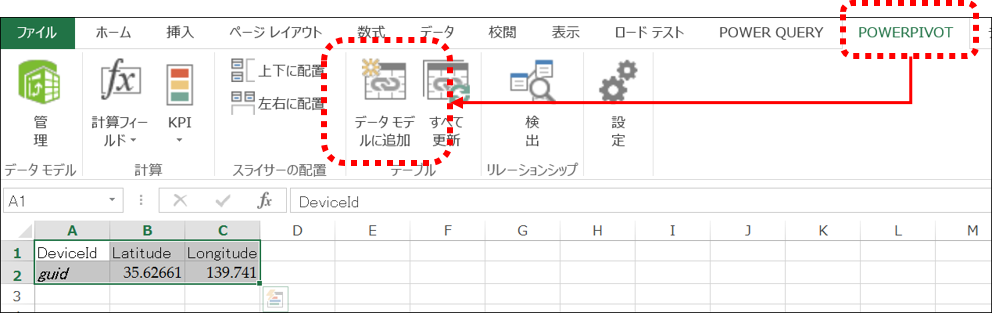
先ず、<http://www.geocoding.jp> 等を使って、皆さんのいる場所の緯度経度を調べてください。

ちなみに、日本マイクロソフトの本社は、緯度が35.62661、経度が139.740987です。

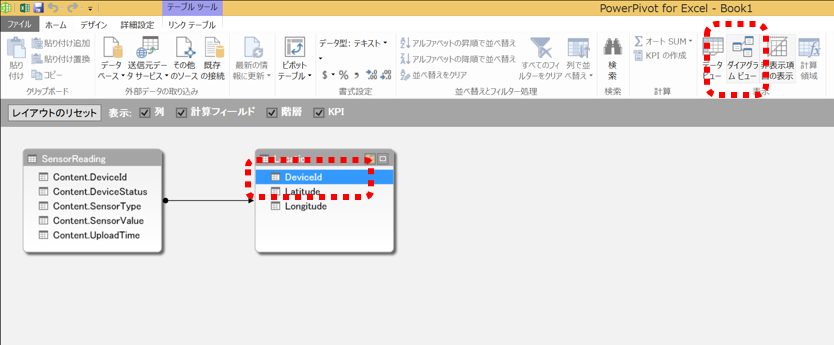
調べた緯度経度をExcelのシートに入力します。



左端は、“DeviceId”と入力し、”guid”は、Step1接続編で作成したIoT Kit向けに作成したGUIDを入力してください。入力したら、A1-C2の範囲を選択します。



選択したままで、“POWERPIVOT”タブを選択し、“データモデルに追加”をクリックします。



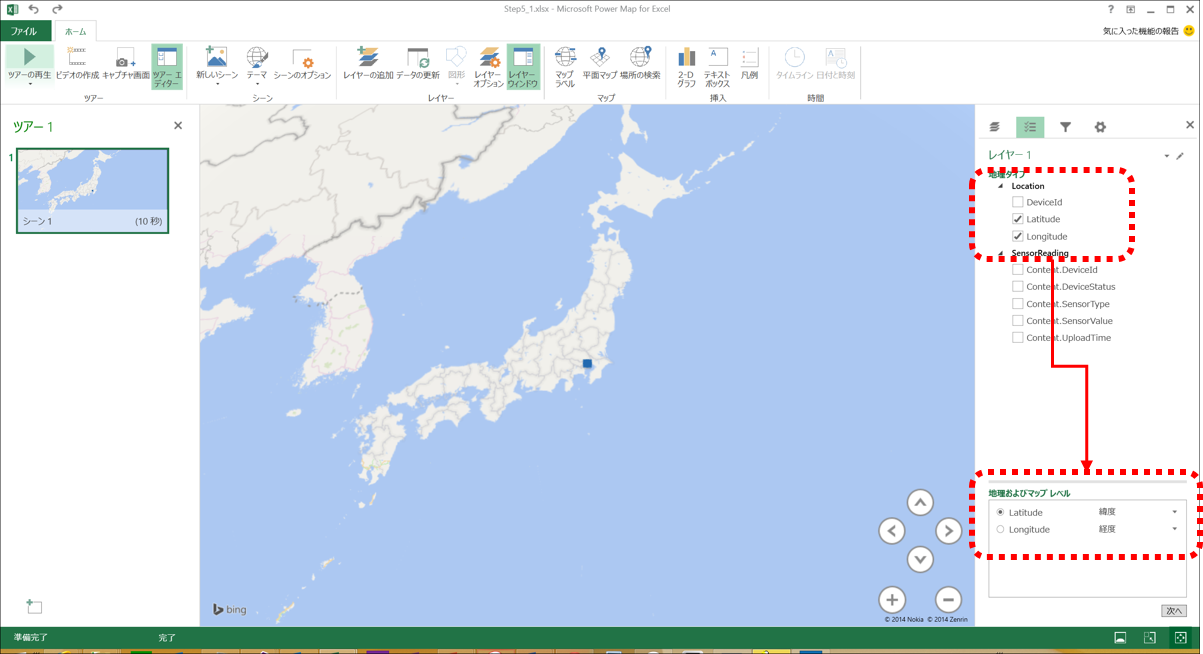
表示されたシートの“ダイアグラムビュー”をクリックすると、上の図のようなデータグラムが表示されます。右側のテーブルの名前は“Location”に変えてありますが、変更しなくても構いません。こちら側の“DeviceId”をクリックし、そのまま“SensorReading”の“Content.DeviceId”にカーソルを移動して、放します。SensorReadingテーブルには位置情報は入っていないので、それに対して位置情報を関連付けたわけです。IoT Kitハードうウェアが複数ある場合は、その数だけLatitude、Longitudeを表に追加してやればマップ化することができます。

位置情報:

Gadgeteer対応のGPSモジュールもあるので、それをIoT Kitに接続し位置情報をGPSから取得するのもまた一興

これで、Power Mapを使う準備が出来ました。

Excelの“挿入”タブをクリックし、“マップ”→“Power Mapの起動”を選択します。



表示されたシートの右側の“Latitude”、“Longitude”にチェックを入れます。すると、下の“地理およびマップレベル”に追加されるので、“Latitude”→“緯度”、“Longitude”→”経度“を選択して”次へ“をクリックします。

次は表示する値を設定します。



“Content.SensorValue”を下の“高さ”にドラッグ＆ドロップし、“Content.UpdateTime”を”時間“にドラッグ＆ドロップします。これで、温度を高さで位置にマップして表示するPower Mapが出来上がりました。地図の下のスライダーの左の”▶“をクリックすれば、棒の高さが温度の値に応じて変わります。またこの地図は、Bing Mapなので、拡大したり前後に移動したり回転したりできるので、色々と試して堪能してください。

Power Mapの詳細は、

<https://support.office.com/ja-jp/article/Power-Map-for-Excel-82d65bd7-70c9-48a3-8356-6b0e82472d74?ui=ja-JP&rs=ja-JP&ad=JP>

をご覧ください。

※今後、品川の日本マイクロソフトオフィスで常時稼働しているIoT Kitのデータを取り込む実習を追加しますので、乞うご期待

|  |
| --- |
| 1. Stream Analyticsによる分析 |
| このSTEPでは、Stream Analyticsの基礎を学びます。   * **IoT Kitのコード修正** * **SQL Azureによる出力用データベース作成** * **Stream Analyticsサービス作成** * **分析の実施** |

## IoT Kitのコード修正

先ず、Stream Analyticsを実行するための入力用Event Hubを作成します。Step 1接続編2.1．の説明を参考に、新しくEvent Hubを作成してください。

そして、Step 2計測編2.2.の学習で作成したIoT KitハードウェアのプログラムのEvent Hubを指定している部分（eventHubName,amqpAddress）を、今、作成したEvent Hubに合わせて変更します。

Stream Analyticsへは、JSON形式でデータを送信します。Uploadメソッドを改造します。

void uploadTimer\_Tick(GT.Timer timer)

{

uploadTimer.Stop();

lock (this)

{

var th = tempHumidSI70.TakeMeasurement();

var accel=accelG248.GetAcceleration();

var message = new Message();

message.Properties = new Amqp.Framing.Properties()

{

Subject = "iotkit",

CreationTime = DateTime.UtcNow

};

message.ApplicationProperties = new Amqp.Framing.ApplicationProperties();

message.ApplicationProperties["time"] =

message.Properties.CreationTime.ToString(

"yyyy'-'MM'-'dd'T'HH':'mm':'ss'.'fffK");

var body = new EGIoTKit.Utility.SimpleJSON();

body.Add("temperature", th.Temperature);

body.Add("accelx", accel.X);

body.Add("accely", accel.Y);

body.Add("accelz", accel.Z);

body.Add("time",

message.Properties.CreationTime.ToString(

"yyyy'-'MM'-'dd'T'HH':'mm':'ss'.'fffK"));

body.Add("deviceid", deviceId.ToString());

body.Add("devicename", deviceName);

message.Properties.ContentType = "text/json";

message.Body = new Data()

{

Binary = System.Text.Encoding.UTF8.GetBytes(body.GetResult())

};

sender.Send(message, SendOutcome, null);

Debug.Print("Sent message at " + message.Properties.CreationTime.ToString());

}

uploadTimer.Start();

}

JSONはテキストなので、べた書きしてもよいのですが、スペルミスしやすいので、ヘルパーライブラリを使ってください。

これで、IoT Kitハードウェア側の変更は終了です。

Event Hubに送信されるのは、加速度（X、Y、Zの三軸）と温度です。Stream Analyticsは、Event Hubに格納されているデータの任意のタイムフレームへの分析が可能なので、ここまでできたら、早速IoT Kitハードウェアにアプリケーションをデプロイして実行し、Event Hubにデータを溜め込みましょう。温度センサーを触れたり触ったり、息を吹きかけたり、IoT Kitハードウェアを小刻みに動かしたりすると、センサー計測値に変化が出て、後で行う分析で特徴値が出やすくなります。色々といじっている様を動画で記録して、後ほど、その画像でいつ何をやっていたか比較しながら、分析結果を見るのもよいでしょう。

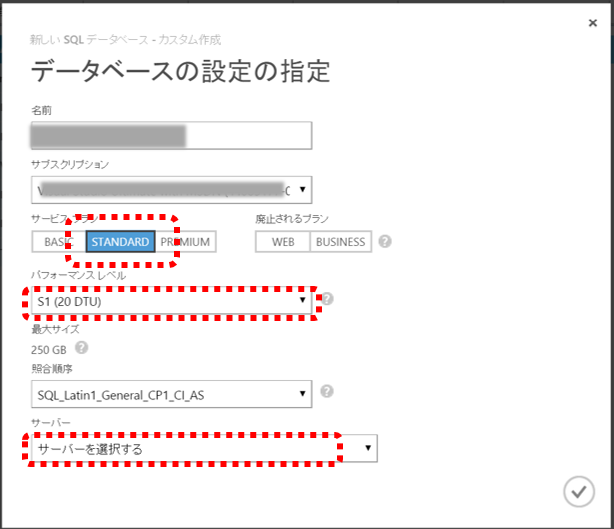
## Azure SQLによる出力用データベース作成

次に、Stream Analyticsを適用した結果を格納するSQLのデータベースを作成します。

Azureポータルで、“SQL データベース”を選択し、左下の“＋新規”をクリックし、左から3番目のカラムで“カスタム作成”を選択します。

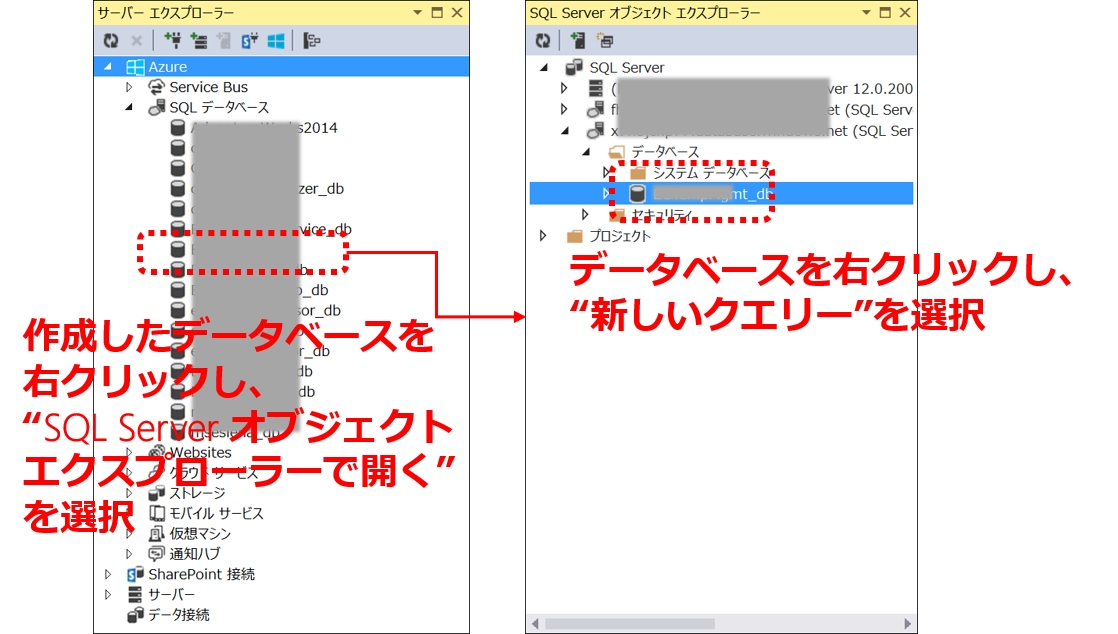
”カスタム作成”をクリックすると、以下のダイアログが表示されます。

以降の実習内容を考慮して、サービスプランは“STANDARD”、パフォーマンスレベルは“S1(20 DTU)”を選択しておきましょう。

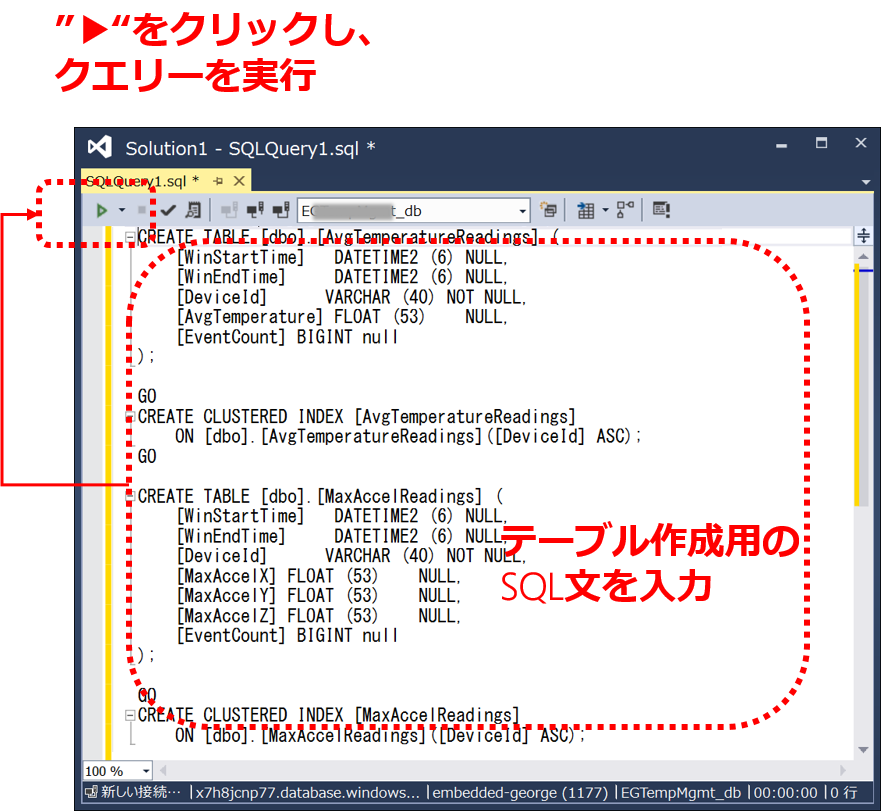
“サーバー”は、新規に作成、Step 4表示編で作成したサーバーを使用する、どちらでも構いません。

データベースの作成方法については、Step 4 表示編の2.1 Mobile Serviceの作成を参考にしてください。

次に、Stream Analyticsの分析結果を格納するテーブルを作成します。サーバエクスプローラーで、Azureを展開します。



一つ前のステップで作成したデータベースを右クリックし、コンテキストメニューで、“SQL Serverオブジェクトエクスプローラーで開く”を選択します。途中にファイアウォール設定などありますが、適宜入力してください。そして、開いたビューで、データベースを右クリックし、コンテキストメニューで“新しいクエリー”を選択します。



表示されたSQLクエリエディターにStream Analyticsの結果を格納するテーブルを作成する、クエリーを入力し、実行します。入力するクエリーは以下の通りです。

CREATE TABLE [dbo].[AvgTemperatureReadings] (

[WinStartTime] DATETIME2 (6) NULL,

[WinEndTime] DATETIME2 (6) NULL,

[DeviceId] VARCHAR (40) NOT NULL,

[AvgTemperature] FLOAT (53) NULL,

[EventCount] BIGINT null

);

GO

CREATE CLUSTERED INDEX [AvgTemperatureReadings]

ON [dbo].[AvgTemperatureReadings]([DeviceId] ASC);

GO

CREATE TABLE [dbo].[MaxAccelReadings] (

[WinStartTime] DATETIME2 (6) NULL,

[WinEndTime] DATETIME2 (6) NULL,

[DeviceId] VARCHAR (40) NOT NULL,

[MaxAccelX] FLOAT (53) NULL,

[MaxAccelY] FLOAT (53) NULL,

[MaxAccelZ] FLOAT (53) NULL,

[EventCount] BIGINT null

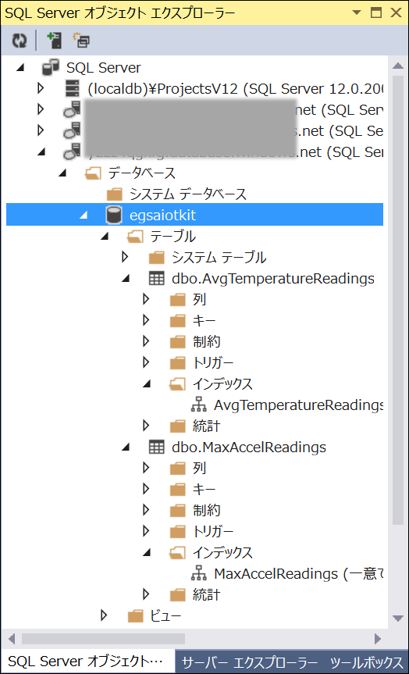
);

GO

CREATE CLUSTERED INDEX [MaxAccelReadings]

ON [dbo].[MaxAccelReadings]([DeviceId] ASC);

最初のCREATE TABLE文は、温度の平均値を格納するためのテーブルを作成しています。次のCREATE TABLE文は、加速度の最大値を格納するためのテーブルを作成しています。オブジェクトエクスプローラーで表示すると、下図の様になっていればOKです。※データベース名は違っていても構いません。

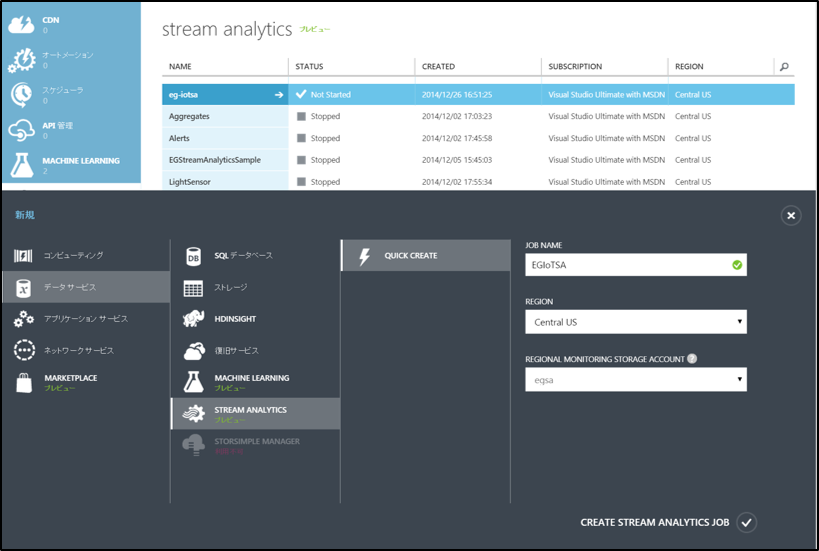


これで、出力を保持するデータベースの準備が終わりました。

## Stream Analyticsサービス作成

前のステップまでで、入力用のEvent Hubと出力用のSQL データベースが作成され、準備が出来たので、Stream Analyticsサービスを作成します。最初は、温度の平均値を抽出するStream Analyticsを作成します。

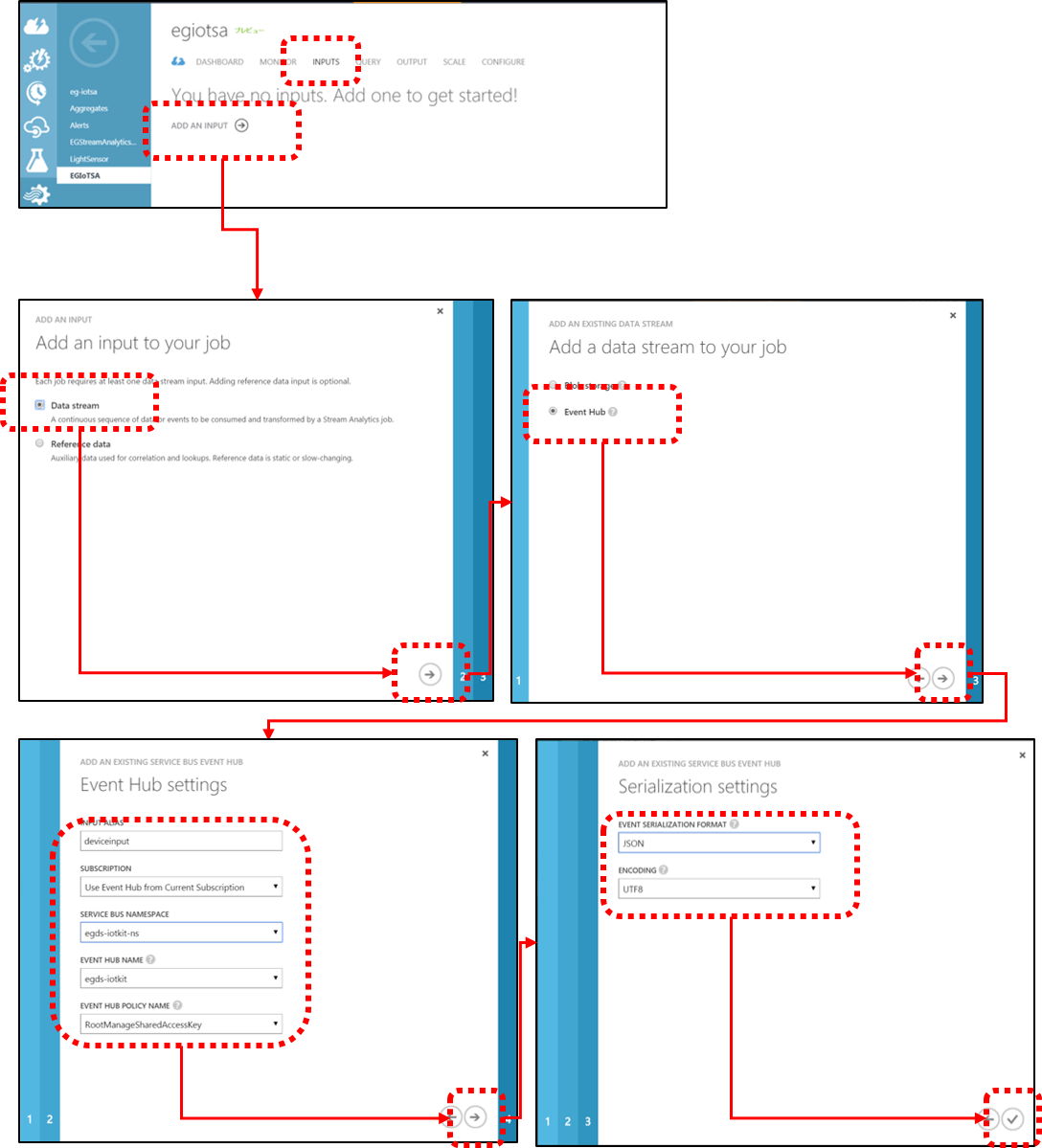
Azureポータルで、“Stream Analytics”を選択し、“＋新規”ボタンをクリックします。



…と、“REGIONAL MONITORING STORAGE ACCOUNT”という項目が出てきました。逸る気持ちを押さえて、一旦、右の×ボタンでこのダイアログを閉じ、Step3蓄積編1.2で説明した方法に従って、Stream AnalyticsのREGIONと同じREGIONで、一つストレージアカウントを作ってください。そしてまたStream Analyticsの新規作成に戻ってください。

### 入力の定義

作成したStream Analyticsを選択して、“INPUT”タブをクリックします。



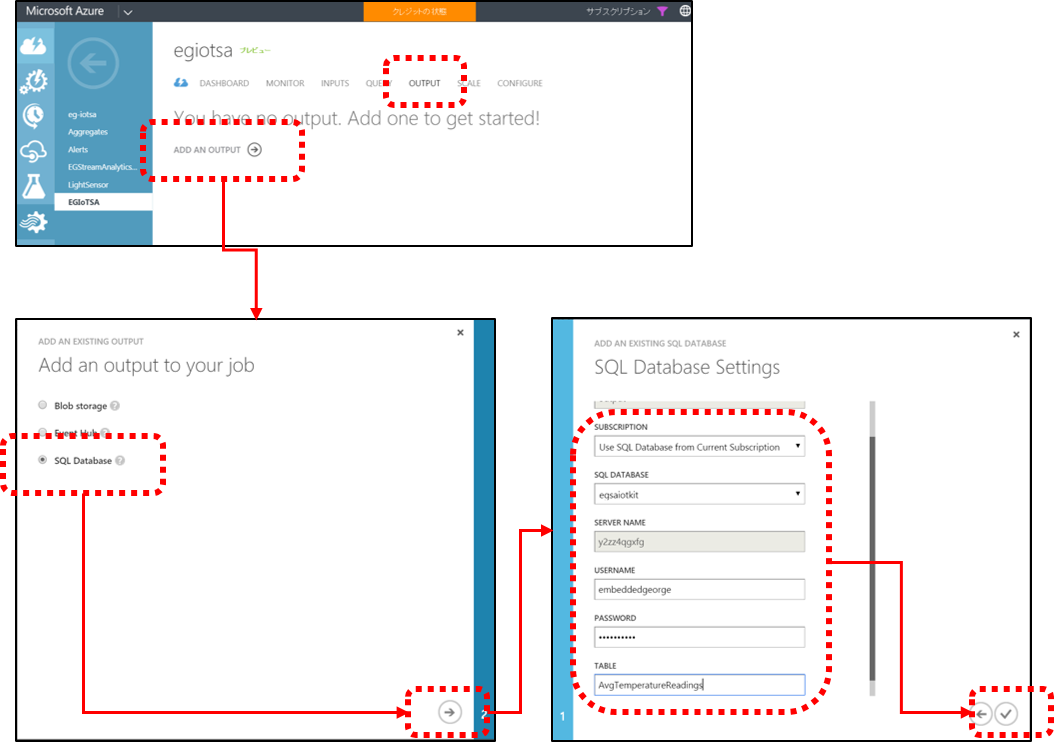
“ADD AN INPUT”をクリックし、“Data Stream”→“Event Hub”と選択し、先ほど作成したEvent Hubを選択してください。“Event Hub Settings”のページでは、Aliasを“deviceinput”と入力してください。最後の”Serialization settings”では、“JSON”、“UTF8”を選択します。IoT Kitハードウェア側の送信は、JSON形式でしたね。

入力の設定はこれで完了です。

### 出力の定義

作成したStream Analyticsの“OUTPUT”タブをクリックします。

ここで、先ほど作成したSQLデータベースを指定します。

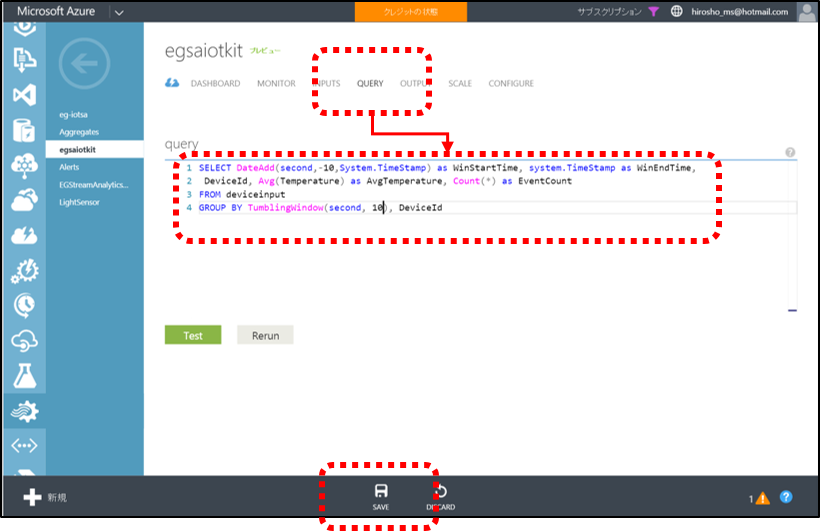


“Add an output to your job”で”SQL Database“を選択します。そして、次のページの”SQL Database Settings“で、”SUBSCRIPTION“を”Use SQL Database from Current Subscription“を選択します。そして、前のステップで作成したデータベース名を”SQL DATABASE“のコンボボックスから選択し、”USERNAME“、”PASSWORD”に、それぞれ、データベースを作成した時に入力したユーザー名とパスワードを入力します。先ずは、温度センサーの計測値の平均値を抽出するので、“TABLE”に“AvgTemperatureReading”と入力します。

### クエリーの定義

温度センサーの計測値の平均値を抽出するクエリーを定義します。

先ず、Stream Anayticsの“QUERY”をクリックします。



“query”に、以下のステートメントを入力します。

SELECT DateAdd(second,-10,System.TimeStamp) as WinStartTime, system.TimeStamp as WinEndTime,

DeviceId, Avg(Temperature) as AvgTemperature, Count(\*) as EventCount

FROM deviceinput

GROUP BY TumblingWindow(second, 10), DeviceId

これは、Event Hubに次々と送られてくる、IoT Kitの温度センサーのデータストリームを、10秒のタイムフレームで、平均値をとっていきます。WinStartTime、WinEndTimeはそれぞれ、AvgTemperature（温度の平均値）のタイムフレームの開始時間と終了時間です。EventCountは、そのタイムフレームに含まれるメッセージの数です。これらは、前のステップで作成したSQLデータベースのテーブルで定義していたカラムに対応しています。また、”FROM deviceinput”の“deviceinput”は、Stream Analyticsの”INPUT“でEvent Hubを指定した際に入力した、”alias”です。

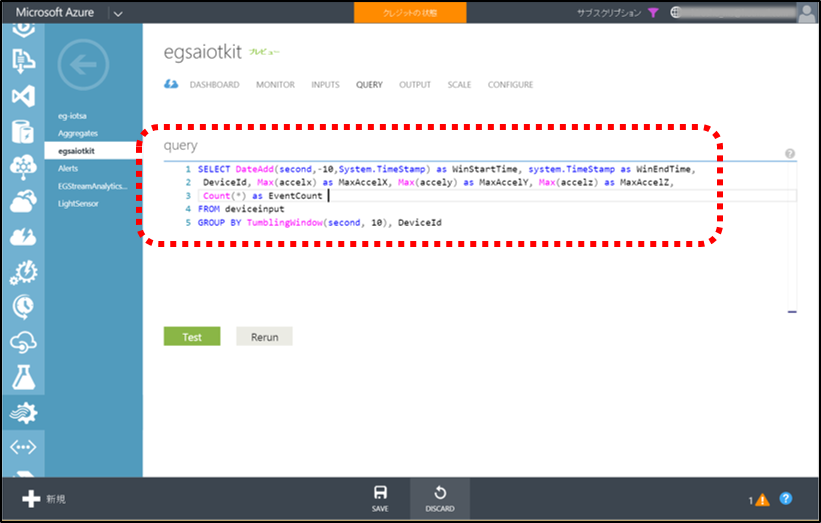
入力したら、“SAVE”をクリックして保存します。

これで、温度センサーに関するStream Analyticsの作成が終わりました。

次に、このステップで学習した方法を基に、加速度センサーの最大値を抽出するStream Analyticsを作成します。これまでの説明を読み返し、もう一つ新しくStream Anayticsを作成し、“INPUT”は同じEvent Hubを選択、そして、“OUTPUT”に、前のステップで作成した“MaxAccelReading”テーブルを設定します。



“OUTPUT”を設定した後は、“QUERY”をクリックして、



次の様に“query”に入力してください。

SELECT DateAdd(second,-10,System.TimeStamp) as WinStartTime, system.TimeStamp as WinEndTime,

DeviceId, Max(accelx) as MaxAccelX, Max(accely) as MaxAccelY, Max(accelz) as MaxAccelZ,

Count(\*) as EventCount

FROM deviceinput

GROUP BY TumblingWindow(second, 10), DeviceId

ここでは、Max関数を使って、10秒間のタイムフレームでの、X軸、Y軸、Z軸ぞれぞれの方向の加速度の最大値を取り出します。

“SAVE”をクリックして、こちらのStream Anayticsも完成です。

クエリーの詳しい記述方法は、

<http://azure.microsoft.com/ja-jp/documentation/services/stream-analytics/>

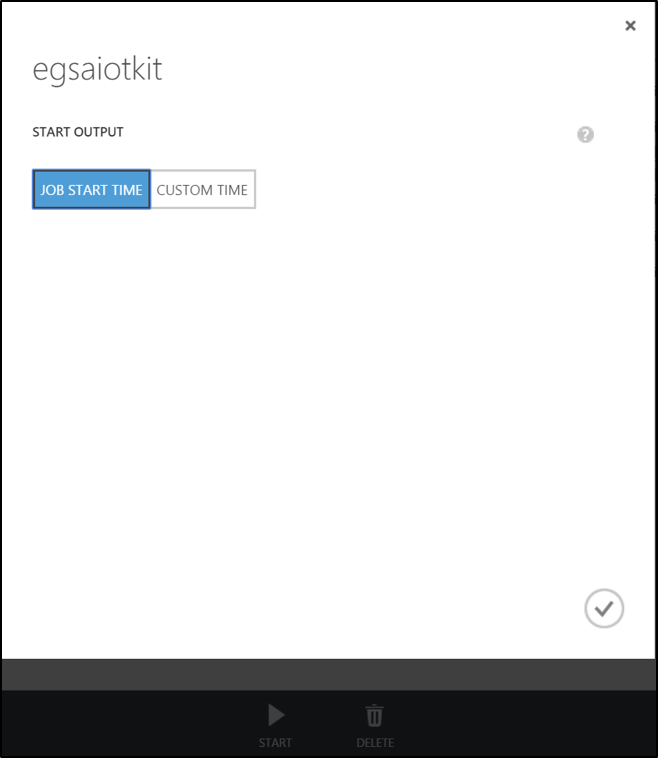
を参考にしてください。

## 分析の実施

さて、“Stream Analytics”のリストページで、実行したいStream Analyticsを選択します。（名前の右の方をクリックします）

下の黒帯の“▶START”をクリックしてください。

すると、（※名前は各自が作成した名前になります）



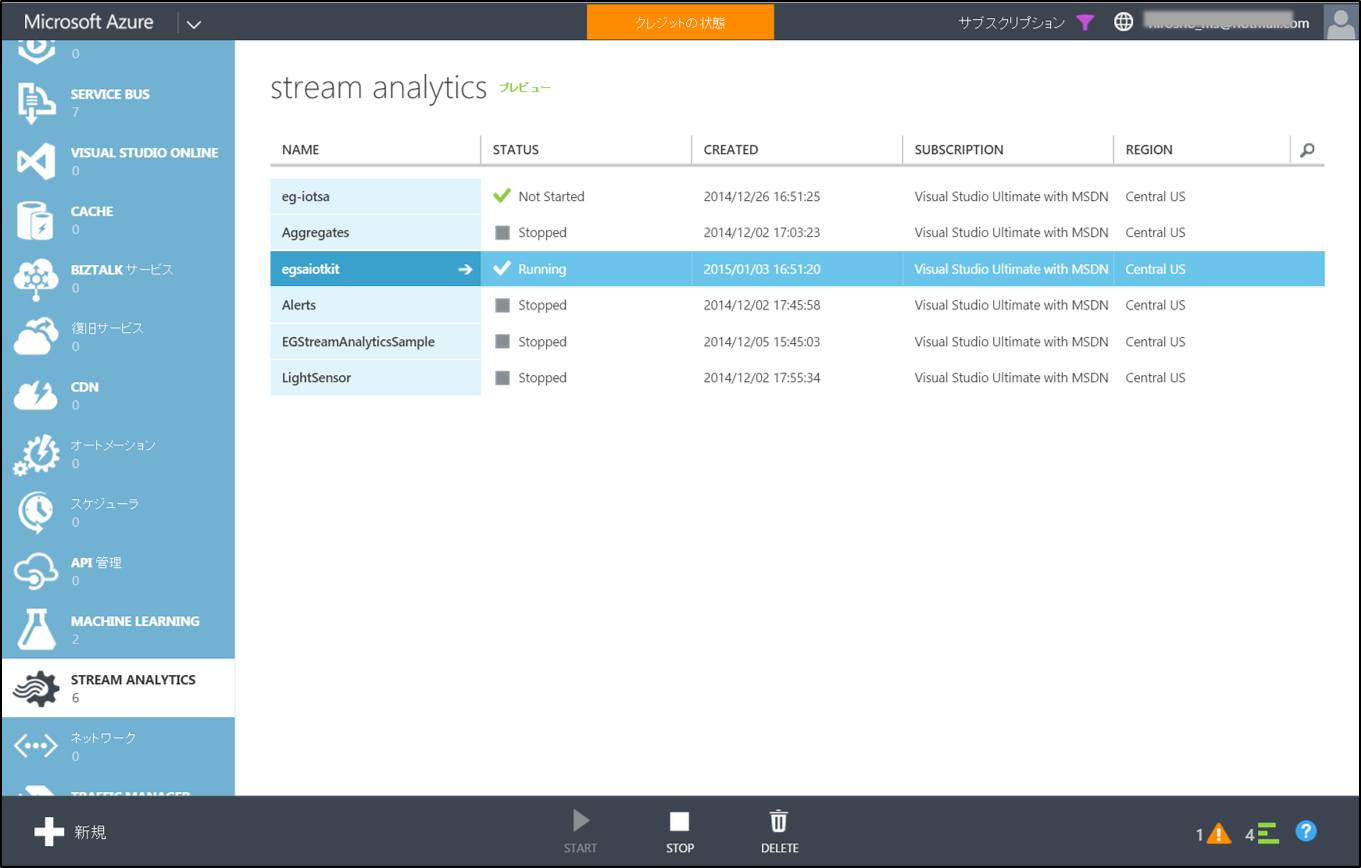
“JOB START TIME”を選択したまま、右下の✓ボタンをクリックすると、現在時刻からの分析が開始されます。この状態で、IoT Kit ハードウェアのアプリを動かして、センサーの値をEvent Hubに送信します。

“CUSTOM TIME”を選択すると、



このように、分析を開始する時間を指定することができます。

どちらのケースも、✓ボタンをクリックしてしばらくすると、Stream Analyticsサービスが起動され、以下のように“Running”という状態に変わります。



サービスを止めたい場合は、下の黒帯の“■STOP”をクリックしてください。

平均温度、加速度の最大値それぞれのStream Analyticsを起動し、IoT Kitハードウェアのアプリを動かして、温度、加速度をEvent Hubに送信してください。

しばらくたったら、分析結果を見てみましょう。Visual Studioの“SQL Serverオブジェクトエクスプローラー”で、“AvgTemperatureReading”、“MaxAccelReading”テーブルをそれぞれ右クリックし、コンテキストメニューで“データの表示”を選択すると、Stream Analyticsが蓄積したデータが表示されます。

前のステップ5.1で実習したPower Queryは、SQLデータベースのテーブルデータをダウンロードすることも可能です。取り出して、Power ViewやPower Mapで可視化してみましょう。

|  |
| --- |
| 1. Machne Learningによる分析 |
| このSTEPでは、Machine Learningの使い方の基礎を実習します。   * **Machine Learning作成** * **活用アプリ作成** |

## Machine Learning作成

このステップではML（Machine Learning）を利用する方法を実践します。

と、その前に、MLを利用するには先ず、何を機械学習するかシナリオが必要です。ここでは、IoT Kitハードウェアに装備されている加速度センサーと温度センサーを活用します。

### Machine Learningシナリオ

IoT Kitハードウェア上で、加速度、温度と、Step 4表示編で実践した、温度が30度以下なら“cool”、30度より上なら”hot”というデバイスの状態をAzureに送信するアプリを動かします。

アプリが動いた状態で、IoT Kitハードウェアの温度センサーを指で触りながら、加速度センサーの計測値が大きく変わるようにダイナミックに（ぶつけて壊さないようにね）IoT Kitハードウェアそのものを動かします。しばらく動かしたら、机などに置いて、温度センサーから指を離します。これを数回繰り返してください。Azureに送信されたIoT Kitハードウェアのデータは、ML用のテーブルに蓄積しておきます。蓄積されたデータは、最初IoT Kitハードウェアが静止状態なので、加速度の値は地球の方向に1Gで、温度は室温です。温度センサーに触ってIoT Kitハードウェアを動かすと、温度が徐々に上がって指の温度（34度ぐらい）に達して定常状態に達しIoT Kitを机等に置くまで、加速度はIoT Kitハードウェアのダイナミックな動きに応じた値が記録されます。机等にIoT Kitを置くと、温度センサーは徐々に室温に戻り、加速度はまた地球の方向に1Gの値になります。センサーの誤差や微妙な振動によりホワイトノイズ的な揺らぎとともにデータはAzureのテーブルに格納されています。この間、デバイス状態は、温度センサーの値に従って、“cool”→”hot”→“cool”に変化します。

蓄積データの加速度だけを見れば、加速度がドラスティックに変わりだすと、デバイスの状態が“cool”→”hot”に変わるように見えますね。

ということで、このステップでは、加速度だけからデバイスの状態を予測する学習モデルを作成することとします。テーブルに蓄積した加速度、温度、デバイス状態のデータは学習用の正解データとして使用します。

### IoT Kitハードウェアのアプリの変更

先ず、Step 4表示編の1.1．で作成したIoT Kitの送信コードを以下の様に変更します。

void uploadTimer\_Tick(GT.Timer timer)

{

uploadTimer.Stop();

var request = HttpWebRequest.Create("http://*[IoTCloud]*.azurewebsites.net/api/**SensorML**") as HttpWebRequest;

request.Headers.Add("device-id", deviceId.ToString());

lock (this)

{

**request.Headers.Add("temperature", lastTemperature.ToString());**

**request.Headers.Add("accel-x", lastAccelX.ToString());**

**request.Headers.Add("accel-y", lastAccelY.ToString());**

**request.Headers.Add("accel-z", lastAccelZ.ToString());**

request.Headers.Add("device-status", deviceStatus);

}

using (var response = request.GetResponse() as HttpWebResponse)

{

if (response.StatusCode == HttpStatusCode.OK)

{

var reader = new StreamReader(response.GetResponseStream());

string message = reader.ReadToEnd();

Debug.Print(message);

}

}

uploadTimer.Start();

}

温度、加速度の全てを送るように変更されました。

### 受信用Webアプリの変更

Step 4 表示編1.2．で改造した“*IoTCloud*”プロジェクトの“Controllers”フォルダーに“SensorMLController”という名前で、コントローラクラスを追加し、以下の様にコーディングします。

public class SensorMLController : ApiController

{

public async void Get()

{

var storeCS = CloudConfigurationManager.GetSetting("StorageConnectionString");

var storageAccount = CloudStorageAccount.Parse(storeCS);

var tableClient = storageAccount.CreateCloudTableClient();

var sensorReadingTable = tableClient.GetTableReference("SensorMLReading");

await sensorReadingTable.CreateIfNotExistsAsync();

var now = DateTime.Now;

var header = this.Request.Headers;

var sensorReading = new SensorMLReading(header.GetValues("device-id").ElementAt(0), now)

{

Temperature = Double.Parse(header.GetValues("temperature").ElementAt(0)),

AccelX = Double.Parse(header.GetValues("accel-x").ElementAt(0)),

AccelY = Double.Parse(header.GetValues("accel-y").ElementAt(0)),

AccelZ = Double.Parse(header.GetValues("accel-z").ElementAt(0)),

DeviceStatus = header.GetValues("device-status").ElementAt(0)

};

await sensorReadingTable.ExecuteAsync(TableOperation.Insert(sensorReading));

}

}

public class SensorMLReading : TableEntity

{

public SensorMLReading(string deviceId, DateTime time)

{

PartitionKey = deviceId;

RowKey = deviceId + time.Ticks.ToString();

DeviceId = deviceId;

UploadTime = time;

}

public string DeviceId { get; set; }

public double Temperature { get; set; }

public double AccelX { get; set; }

public double AccelY { get; set; }

public double AccelZ { get; set; }

public string DeviceStatus { get; set; }

public DateTime UploadTime { get; set; }

}

Getメソッドで、IoT Kitハードウェアから送信された、温度、加速度、デバイス状態を取り出し、“SensorMLReading”というテーブルに格納します。

SensorMLReadingクラスは、格納用テーブルの定義です。修正が終わったら、“発行”してAzureのWeb Siteを更新します。

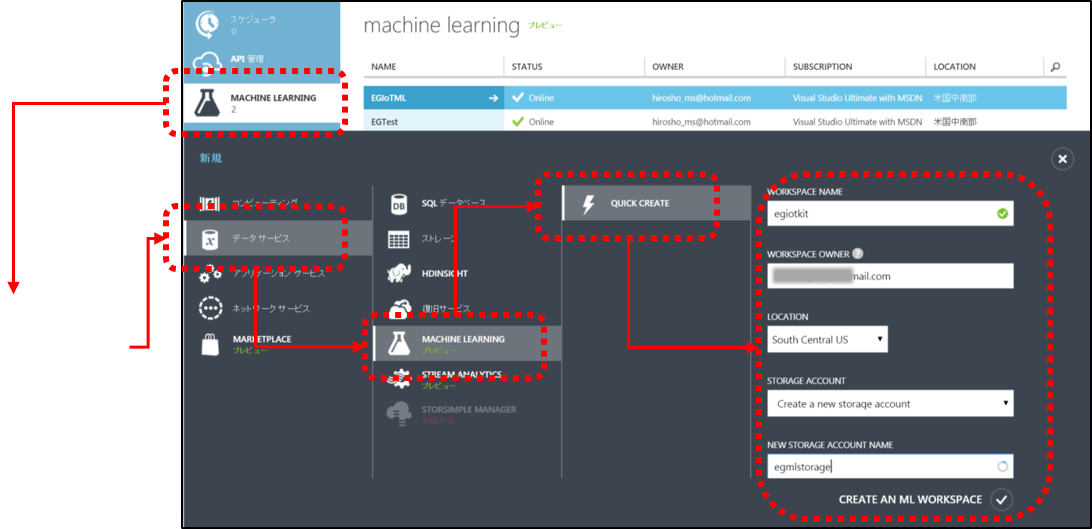
### 学習用データの蓄積

Webサイト側の改造が完了したら、IoT Kitハードウェアのアプリを実行し、冒頭のシナリオに従って、データを蓄積します。長く蓄積すればするほど、学習モデルの精度は高まります。10回ぐらいは温度センサーを指で温めながら動かしたり、温度センサーから指を離して置いたり、を繰り返してください。

データが溜まったら、いよいよMLの作成に取り掛かります。

### Machine Learningの新規作成

Microsoft Azure のポータルを開き、左側の機能で“Machine Learning”を選択し、下の黒帯の左側、“+新規”をクリックし、

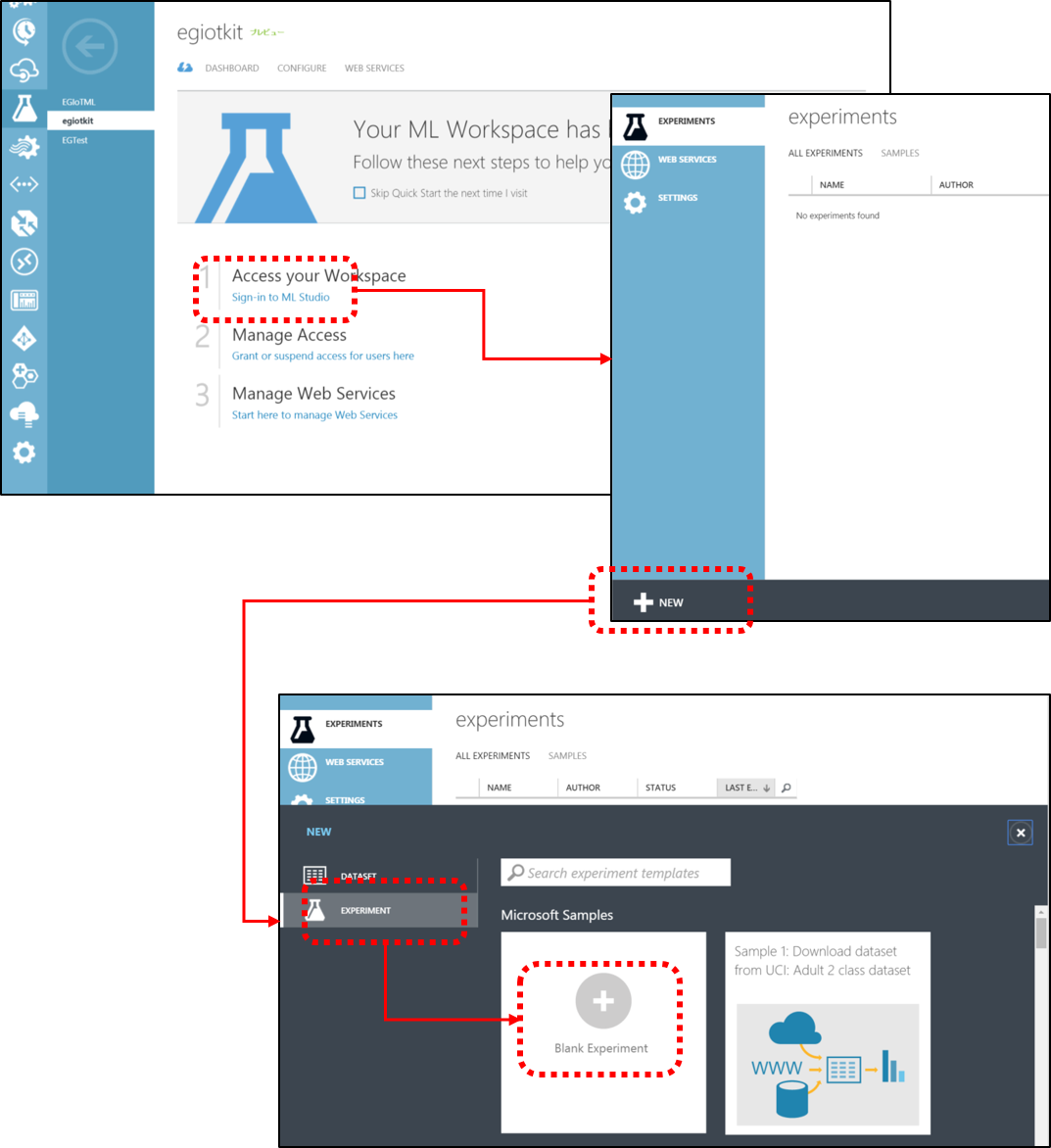


“WORKSPACE NAME”には適当な名前を入力し、“LOCATION”はそのまま（現在プレビューにつき図のロケーションしか選択できません）で、”STORAGE ACCOUNT“は、”Create a new storage account“を選択して、”NEW STORAGE ACCOUNT NAME“にも、適当な名前を入力して、右下の”✓ボタン“をクリックします。新しいMLの作成が完了すると、”machine learning”のリストに追加されます。追加されたMLをクリックしてください。

### 学習モデルの構築

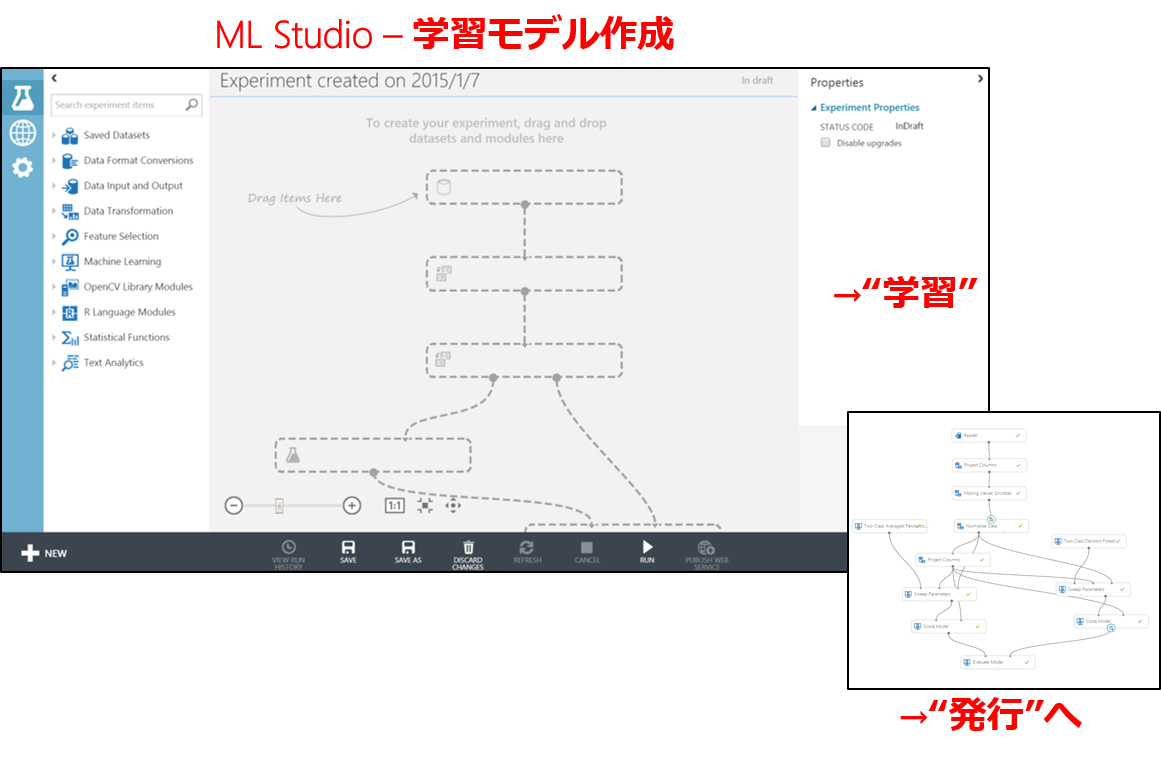
表示されたページで、学習モデルを作っていきます。

“Access Your Workspace”の下の“Sign in to ML Studio”をクリックします。表示されたページの左下の”+ NEW“をクリックします。

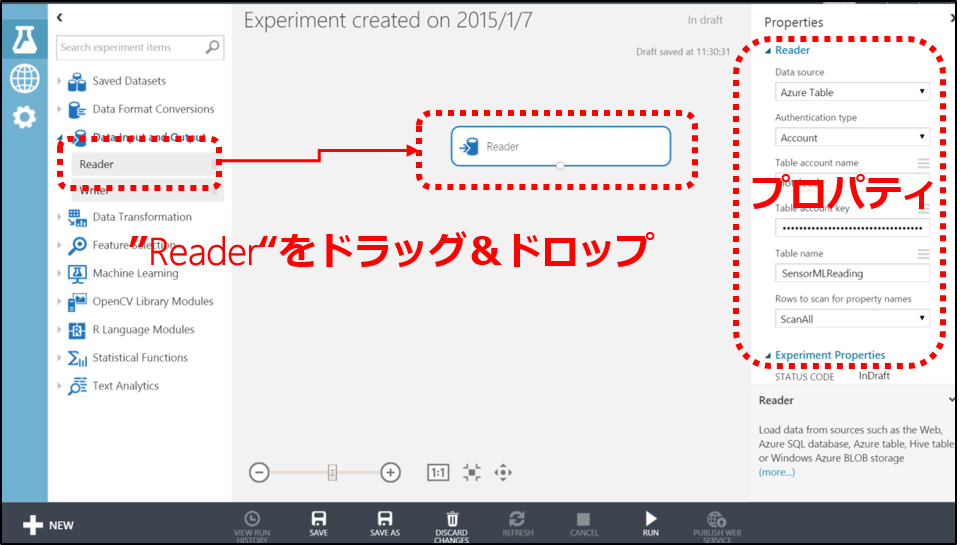


“EXPERIMENTS”を選択し、“Blank Experiment”の”+“をクリックします。すると、学習モデルのエディタが表示されます。この環境で、学習モデルの作成と学習、成長したモデルの発行を行っていきます。ML Studioは、.NET GadgeteerのVisual Studio開発環境のように、左側に学習モデルを構築する部品が用意され、真ん中の灰色の部分で、それら部品をドラッグ＆ドロップし、部品間のデータフロー定義を行います。部品を選択すると、右側に対応するプロパティや設定が表示され、そこで詳細を定義していきます。

ここでは、IoT Kitハードウェアから送られてきたデータを格納している“SensorMLReading”テーブルを入力として、2種類のアルゴリズムの学習を試し、より精度の高い学習モデルを選択して、Web Serviceとして発行します。ML Studioの使い方は、<http://help.azureml.net/Content/html/96b39d63-f6dd-4461-a244-b90ab3891cda.htm> を参考にしてください。



先ずは、“SensorMLReading”テーブルからデータを読み込む“Reader”を定義します。



左のツールボックスの“Data Input and Output”を展開して、”Reader”をマウスでドラッグ＆ドロップします。“Reader”を選択すると、右にこのモジュールのプロパティが表示されます。このモジュールは、IoT Kitのデータが格納されている“SensorMLReading”テーブルのReaderにするので、“Data source”で“Azure Table”を選択し、“Authentication Type”は”Account“に、”Table Account Name“と”Table Account Key“は、”SensorMLReading“テーブルのストレージアカウント名と、アクセスキーを入力します。

“Table name”は、IoT Kitのデータが格納されたテーブルである、”SensorMLReading“と入力します。”Rows to scan for property names”は、“ScanAll”を選択します。

ここまで出来たら、ちゃんと定義が正しくなされているかをチェックしておきます。

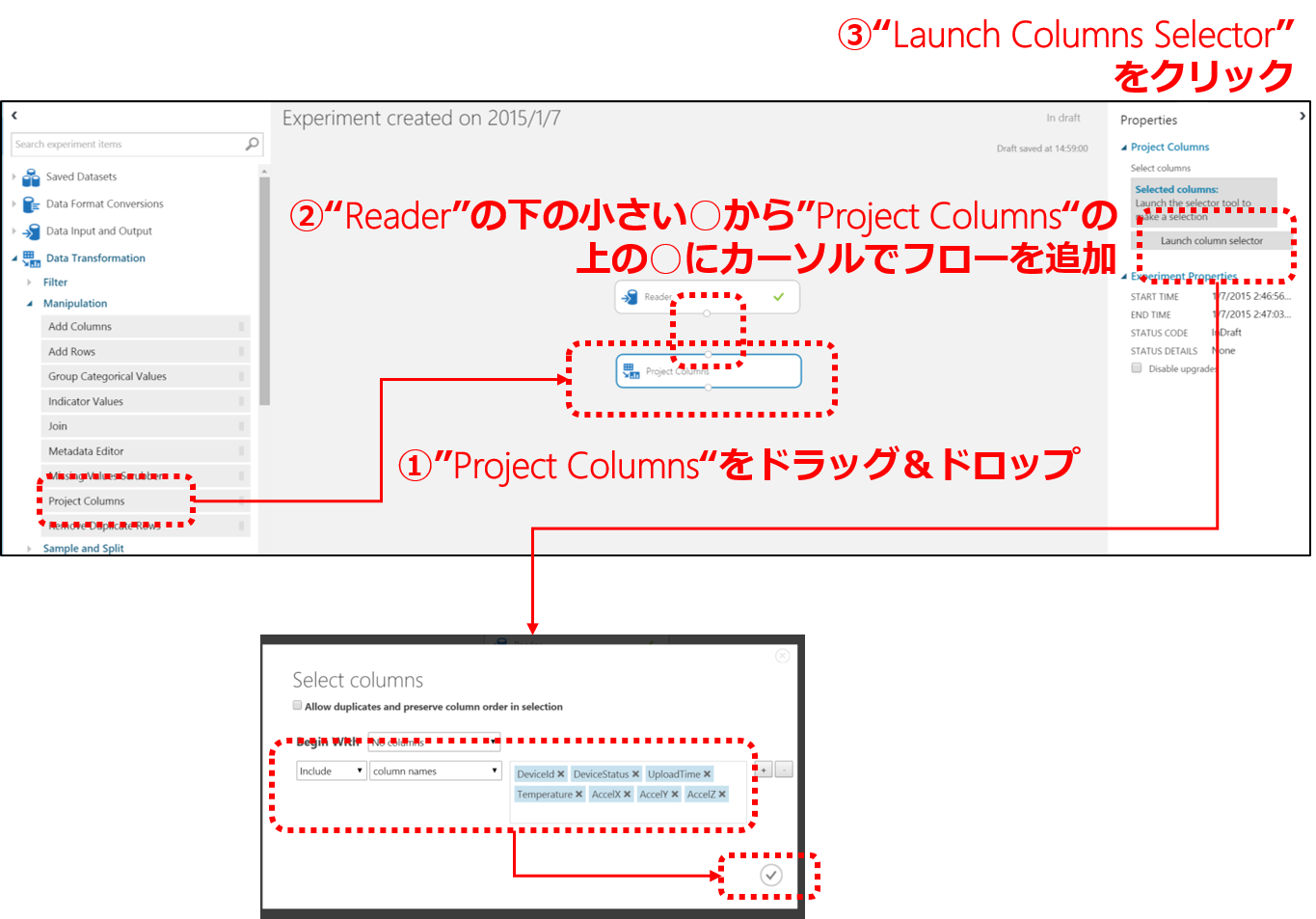
先ず、下の黒帯の“▶RUN”をクリックします。この操作で”SensorMLReading“に蓄積されたデータが読み込まれます。エディタに配置した、“Reader”の下の小さな○を右クリックし、“Visualize”を選択します。



正しく設定されていれば、上の様なダイアログが表示され“SensorMLReading”テーブルに格納されていたデータが表示されます。それらしきデータか、確認してください。

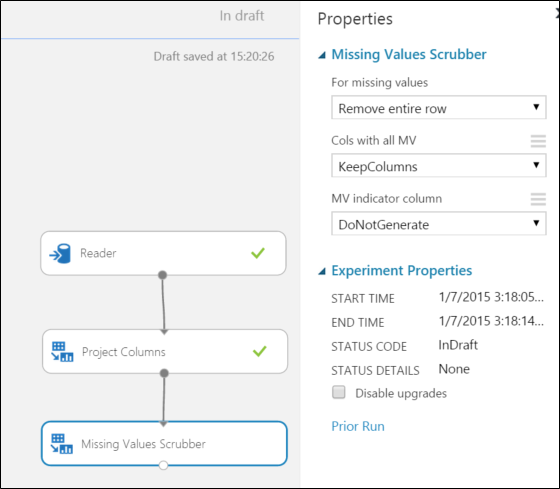
※モジュールの下にある小さな○を右クリック→“Visualize”を選択することにより、対象モジュールの処理結果を参照可能です。以降、モジュールの定義が終わるごとに、“▶RUN”をクリックしてモデルを実行し、実行結果を確認しながら作業を進めてください。何かが間違っていれば、モジュールの右側に赤いバッテンマークが表示されます。

次にReaderを通じて読み込んだテーブルのデータから分析に必要なカラムを抽出するモジュールを追加します。“Data Transformation”→“Manipulation”の“Project Columns”モジュールを追加してください。



“Select columns”では、“Begin With”を“No columns”に設定（初期状態として「何もカラムを選択しない」という意味）し、“include”、”column names”に設定（「右の項目で選択されたカラムを追加する」という意味）します。そして右の欄をクリックするとカラム名の一覧がリスト表示されるので、“DeviceId”、”DeviceStatus“、”UpdateTime”、“Temperature”、“AccelX”、”AccelZ“を選択して、右下の”✓“をクリックします。

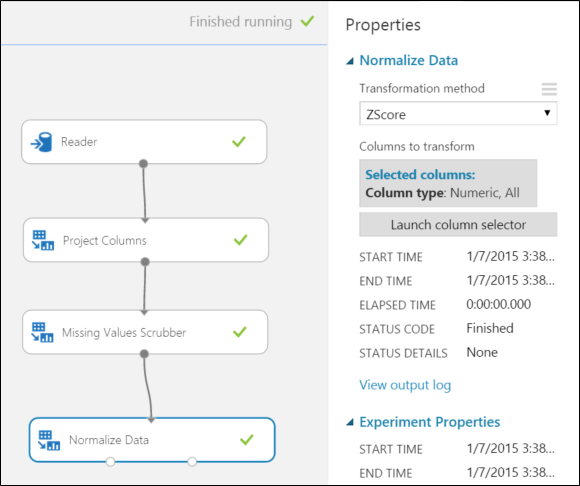
次に、分析をスムーズに行うために、値の入っていないカラムを含む行を削除するフィルターモジュールを追加します。“Data Transformation”→“Manipulation”の”Missing Values Scrubber“モジュールを追加して、”Project Columns“の下の○と追加したモジュールの上の○のフローを作成してください。



“Properties”は、“For missing values”は“Remove entire row”に、“Cols with all MV”は”KeepColumns”を、“MV indicator column”は“DoNotGenerat”で設定します。

次に抽出したカラムの温度と加速度の値を正規化するモジュールを加えます。※このモジュールが必須かどうかは微妙ですが、練習ということで追加してください。

“Data Transformation”→“Scale and Reduce”の“Normalize Data”モジュールを加えます。



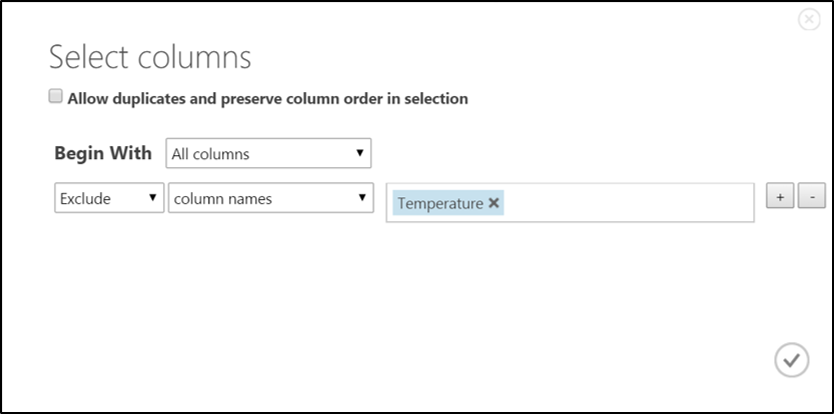
Propertiesは、“Transformation method”を“ZScore”にして、“Columns to transform”の”Launch column selector”をクリックして、“Include”、”column type”、“Numeric”を指定します。数値データは、温度と加速度のデータだけなので、これらのみに正規化が適用されます。

ここまでで、Azureのテーブルに格納されたデータを分析しやすいように加工するフローが出来上がりました。

次に予測モデルの構築と学習させるためのモジュールとフローを追加していきます。

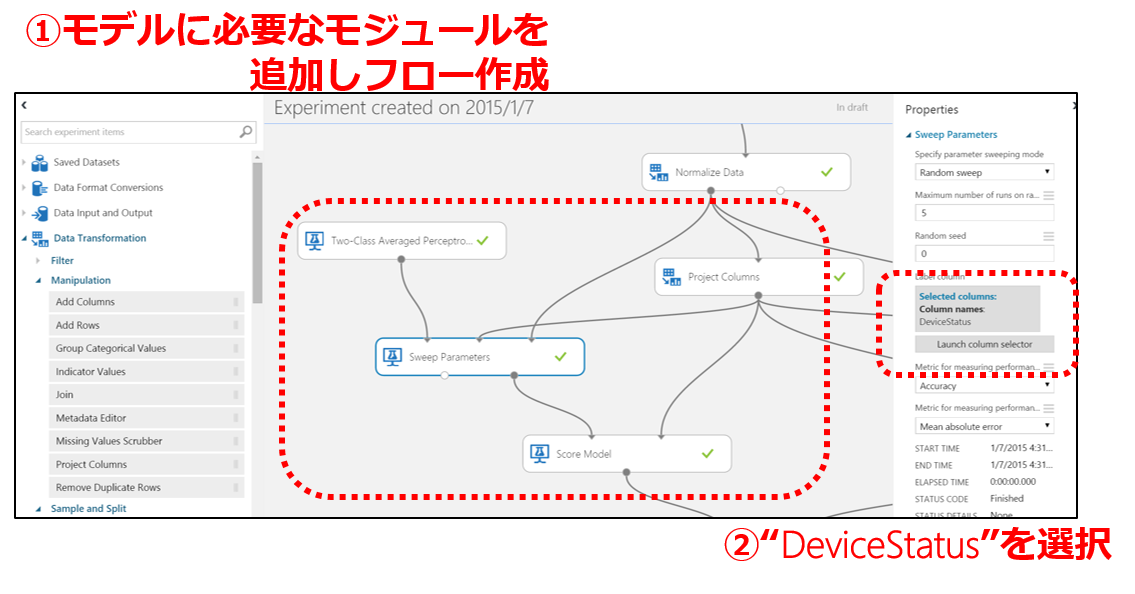
冒頭に説明したシナリオをもう一度ざっくり眺めてみると、本質は、「“AccelX”、“AccelY”、”AccelZ“のデータ傾向から“DeviceStatus”を予測すること」であることがわかります。“Temperature”と“DeviceStatus”は直接的な関係にあるので、学習用データからの排除が必要です。

先ずは、もう一個、“Project Columns”モジュールを追加して、“Normalize Data”モジュールとの間にフローを作成します。そして、“Select columns”で、“Exclude”、”column names”、“Temperature”と設定して“Temperature”のカラムを排除します。



この実習では、2つの予測モデルを作成し、結果を比較します。

まず一つ目の予測モデルを作成します。下図を参考に、“Machine Learning”→“Initialize Model”→“Classification”の”Two-Class Averaged Perceptron“モジュールと、”Machine Learning“→”Train“の”Sweep Parameters“モジュール、そして、”Machine Learning“→”Score“の”Score Model“モジュールを追加して、フローを作成します。



念のため、フローの接続を以下に列記しておきます。

* “Two-Class Averaged Perceptron”モジュール → “Sweep Parameters”モジュールの左端の○

未学習モデルをインプット

* “Project Columns”モジュール → “Sweep Parameters”モジュールの真ん中の○

学習用データ（Temperature抜きのデータセット）をインプット

* “Normalize Data”モジュールの左下の○ → “Sweep Parameters”モジュールの右端の○

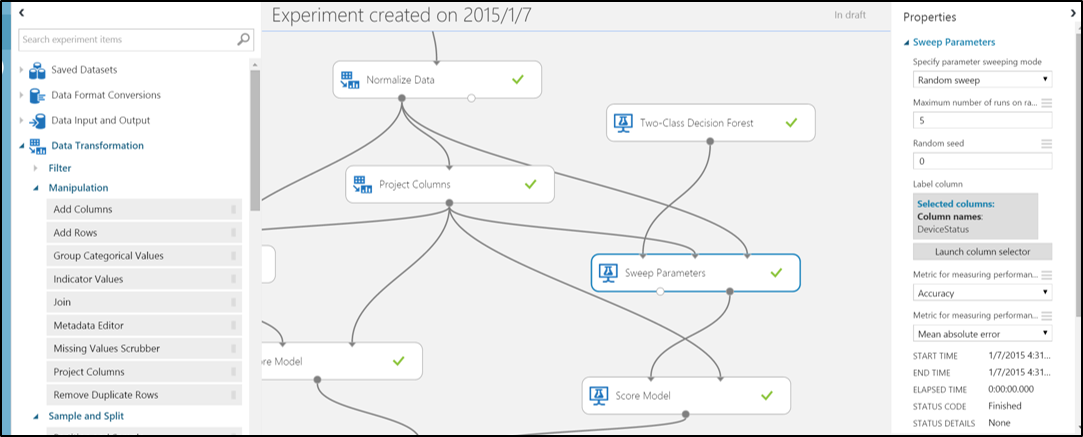
検証用データの（Temperature入りのデータセット）をインプット

* “Sweep Parameter”モジュール → “Score Model”モジュールの左側の○
* “Project Columns”モジュール → ”Score Model“モジュールの右側の○

“Sweep Parameter”モジュールの“Label column”プロパティは、“DeviceStatus”を指定してください。他のプロパティはそのままでOKです。

ここまで出来たら、下の黒帯の“▶RUN”を実行して、一旦学習させてみてください。学習が完了したら、“Score Modelの右下の○を右クリックし、”Visualize“を選択し、”DeviceStatus“と”ScoredLabels”の各行の値を比べてみてください。左と上の軸に近いほど、正確なモデルが出来上がったことになります。

次に、もう一セット、予測モデルを作成します。



こちらのセットでは、“Two-Class Averaged Perceptron”モジュールの代わりに“Two-Class Decision Forest”モジュールを使います。フローは、

* “Two-Class Decision Forest”モジュール → “Sweep Parameters”モジュールの左端の○
* “Project Columns”モジュール → “Sweep Parameters”モジュールの真ん中の○
* “Normalize Data”モジュールの左下の○ → “Sweep Parameters”モジュールの右端の○
* “Sweep Parameter”モジュール → “Score Model”モジュールの左側の○
* “Project Columns”モジュール → ”Score Model“モジュールの右側の○

先ほど作成した予測モデルとの違いは、未学習モデルのモジュールが異なるだけです。 “Sweep Parameters”モジュールの“Label column”を“DeviceStatus”に指定するのをお忘れなく。

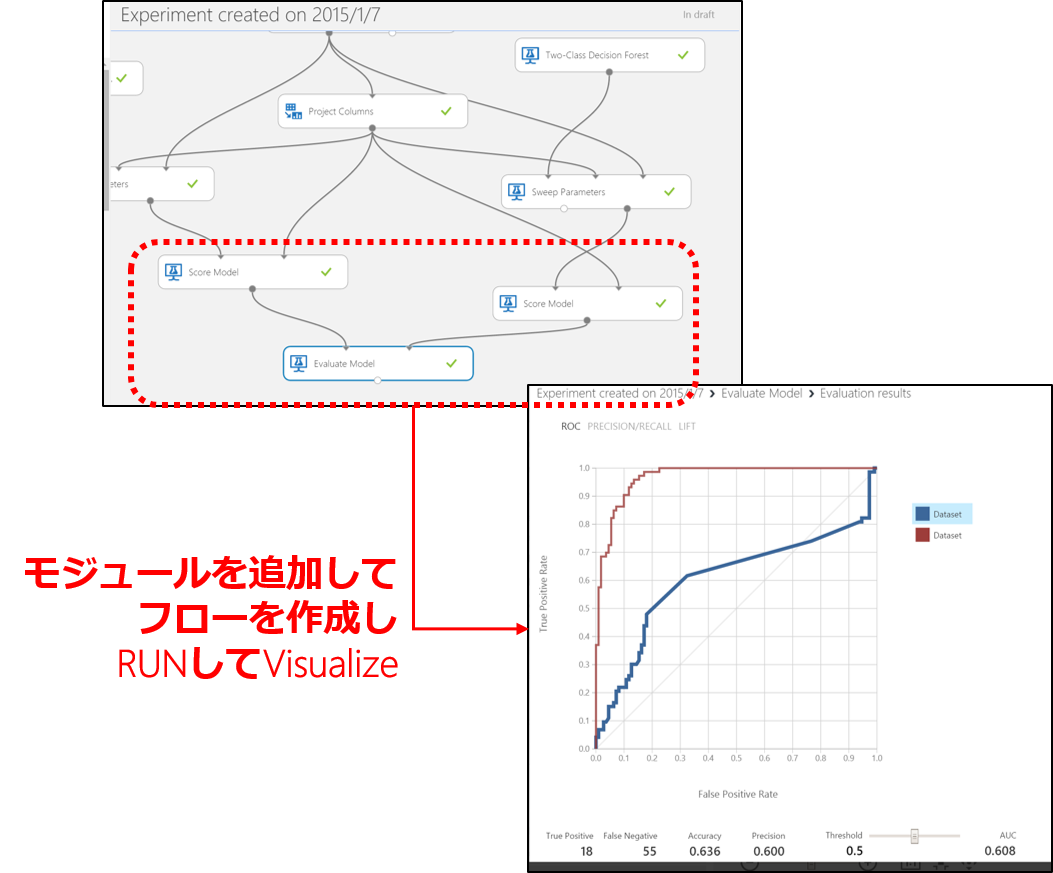
作業が終わったら、“▶RUN”を実行して、“Score Model”の結果を“Visualize”してみます。別の予測モデルの結果と見比べてどちらがより正確に予測しているか確認してみましょう。

二つの予測モデルの作成が終わったら、この二つを比べて評価するモジュールを加えます。

※予測モデルの学習結果は皆さんの環境、IoT Kitの動かし方等、諸々の要因が絡むため、本書記載の状況がそのまま再現されるとは思わないでください。

※“Two-Class Averaged Perceptron”モジュール、“Two-Class Decision Forest”も、学習時に指定するパラメータの設定によって、予測の正確性が変わります。本書ではデフォルトの設定を使っていますが、一通り終わったら、各自、色々とパラメータを振って学習結果が変わることを確認してください。

“Machine Learning”→“Evaluate”の“Evaluate Model”モジュールを追加し、二つの“Score Model”モジュールの下の○からフローを引きます。作業が完了したら、“▶RUN”で実行し、“Evaluate Model”の下の○を右クリック、Visualizeで可視化してみます。

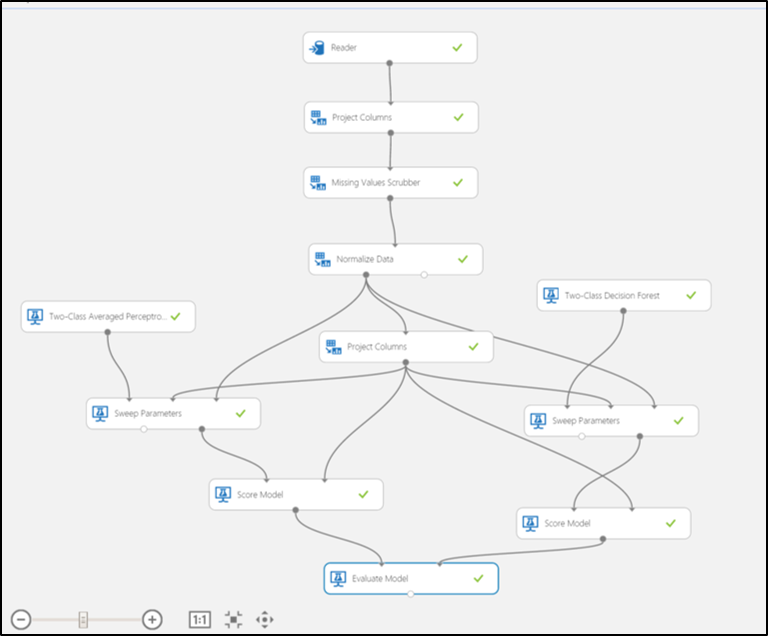


結果は、学習している皆さんのデータによって異なりますが、グラフの左側と上側に近いカーブを描いている方が、より正確な予測をしているモデルであることを意味しています。

* + - この手順書の目的は、あくまでも、ML、及び、ML Studioの使い方の習得の手助けです。使われているモジュールの意味や正しさは、このトレーニングの対象外であり、ここに掲載されたモデルの正しさを保証するものではありません。

以上で、予測モデル、及び、学習の実習は終了です。

参考の為、作成したモデルの全体像を図に示しておきます。また、“▶RUN”で、赤いバッテンが表示されるようでしたら、どこかが間違っているので、再度、“Experiment”を一から作り直し、本書の説明に従ってモデルを作成し直してください。



### 発行

予測モデルが完成したので、IoTシステムで利用できるよう、作成したモデルを“発行”します。

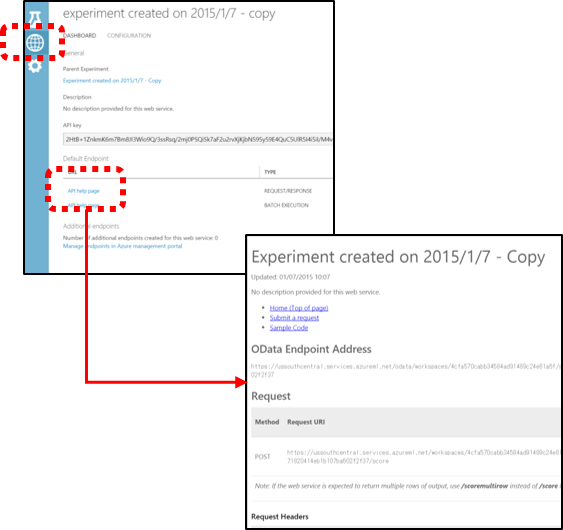
エディタ上で、“Normalize Data”モジュールの上の○を右クリックし“Set as Publish Input”を選択します。これで、入力用の設定が終わりました。

次に、右側の“Score Model”の下の○を右クリックし“Set as Publish Output”を選択します。これで、出力用の設定も終わりです。

エディタ上の表示は下図のように、入力と出力がマークされた状態になります。



下の黒帯の“PUBLISH WEB SERVICE”をクリックすると、Azure上でこの予測モデルを利用するためのWebサービスが作成されます。



Azureでは、MLを“Publish”すると、REST APIでアクセスできる逐次利用とバッチ利用の二つのサービスが自動的に作成されます。それぞれの“API help page”リンクをたどると、使い方がサンプルコード付きで説明されています。

Machine Learningについてもっと知りたい、もっと極めたい、という方は、

<http://azure.microsoft.com/ja-jp/documentation/services/machine-learning/>

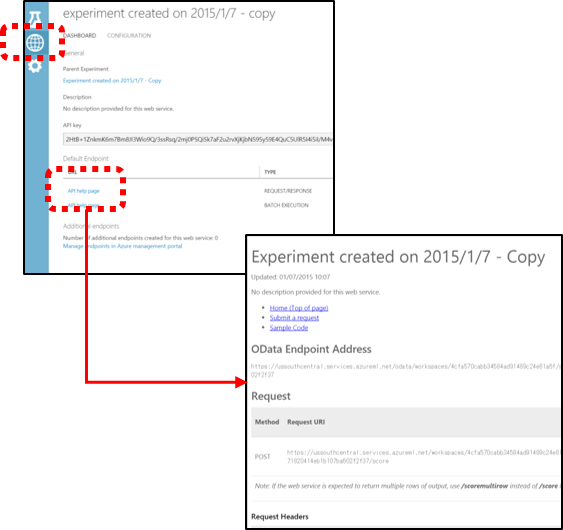
<http://help.azureml.net/Content/html/96b39d63-f6dd-4461-a244-b90ab3891cda.htm>

<https://social.msdn.microsoft.com/forums/azure/en-US/home?forum=MachineLearning>

等を活用してみてください。

## 活用アプリの作成

前節で説明した通り、



Azureでは、MLを“Publish”すると、REST APIでアクセスできる逐次利用とバッチ利用の二つのサービスが自動的に作成されます。それぞれの“API help page”リンクをたどると、使い方がサンプルコード付きで説明されています。ここで紹介されているサンプルは、ストアアプリ（WinRT API）でも、Web Site上のWebアプリでも、IoT Kitハードウェア（.NET Micro Framework：※若干の修正必要）でも埋め込み可能です。Step 1から実習して獲得した知識をフル活用して、発行したMLの予測モデルを組み込んでみてください。

以上で、実習は終了です。

|  |
| --- |
| 1. 試してみよう |
| このステップの学習を完了すると、組込み機器とクラウドの接続、センサー計測、計測データのクラウドへの送信、クラウドからIoT Kitへのコマンド受信、クラウドに送信された計測データの永続ストレージへの蓄積、蓄積したデータの表示、更には分析ができるようになりました。  Step 1~5で学んだ知識を総動員して、センサーの種類を増やしたり、送受信のデータフォーマットやタイミングを変えてみたり、IoT Kitハードウェアを複数接続してみたり、色々なシナリオを想定してシステムを作ってみたり、Excel Power BIでデータ表示を様々に行ってみたり、Stream AnalyticsのQueryで色々とチャレンジしてみたり、MLのモデルを色々と作ってみたり、…と色々とお試しください。  さぁこれで、基本スキルの獲得は完了です。次は応用にチャレンジです。 |