Time Intelligence – YTD

1. Create a new measure in the measures table.
2. Replace the code with the following:

```
YTD Total Sales = TOTALYTD
(
   SUM('Sales OrderDetails'[Order Line Total])
   , Dates[Date].[Date]
)
```

3. Similar to the CALCULATE function, the TOTALYTD function also manipulates the filter context. It creates a running total for the year, no matter which dates, months, quarters, or years have been selected.
4. Add a matrix visual with Month on the rows and Year on the columns. Add YTD Total Sales as the values. It should look like this:

Figure 08-15: TOTALYTD in a measure

5. You can clearly see the total is accumulating as the months progress. You can see it reset when the year resets. The DAX here is deceptively simple, but very powerful! If you wrote something similar in other languages, it would involve a lot of complicated looping and tallying, but DAX does this for you as long as you have a Dates table. The Dates table is necessary for all functions that use time intelligence. There are similar functions for month to date and quarter to date.

Time Intelligence – PREVIOUSMONTH

1. Create a new measure in the measure table.
2. Replace the code to the following:

```
Total Sales Previous Month = CALCULATE
(
   sum('Sales OrderDetails'[Order Line Total])
   , PREVIOUSMONTH(Dates[Date])
)
```

3. This function overrides the filter context to say that no matter which month is selected, give the value for the previous month.
4. Add a Matrix visual with Month as the Row and Year as the Column. Put Total Sales and Total Sales Previous Month in the values. Your visual should look like this:

Year	2006		2007		2008		Total
Month	Total Sales	Total Sales Previous Month	Total Sales	Total Sales Previous Month	Total Sales	Total Sales Previous Month	**Total Sales**
January			$66,692.8	$50,953.4	$100,854.72	$77,476.26	**$167,547.52**
February			$41,207.2	$66,692.8	$104,561.95	$100,854.72	**$145,769.15**
March			$39,979.9	$41,207.2	$109,825.45	$104,561.95	**$149,805.35**
April			$55,699.39	$39,979.9	$134,630.56	$109,825.45	**$190,329.95**
May			$56,823.7	$55,699.39	$19,898.66	$134,630.56	**$76,722.36**
June			$39,088	$56,823.7		$19,898.66	**$39,088**
July	$30,192.1		$55,464.93	$39,088			**$85,657.03**
August	$26,609.4	$30,192.1	$49,981.69	$55,464.93			**$76,591.09**
September	$27,636	$26,609.4	$59,733.02	$49,981.69			**$87,369.02**
October	$41,203.6	$27,636	$70,328.5	$59,733.02			**$111,532.1**
November	$49,704	$41,203.6	$45,913.36	$70,328.5			**$95,617.36**
December	$50,953.4	$49,704	$77,476.26	$45,913.36			**$128,429.66**
Total	**$226,298.5**		**$658,388.75**	**$50,953.4**	**$469,771.34**	**$77,476.26**	**$1,354,458.59**

Figure 08-16: PREVIOUSMONTH in a measure

5. CALCULATE and PREVIOUSMONTH overrides the filter context to display the value for the previous month selected. For instance, in August 2006, you can see that the current value is $26,609.40, while the previous month value is $30,192.10.

X vs Non-X Functions

You may have noticed that there are a lot of functions that have the same name, but are only different because one has an X at the end of it. Some examples are SUM, SUMX, COUNT, COUNTX, MIN, MINX, and so forth. What are the differences between these functions?

The X functions are what are called iterator functions. They are important where the value is important in the context of a row. It works row by row. SUMX has awareness of rows in a table, hence can reference the intersection of each row with any columns in the table. SUMX will add values as it relates to the entire row that is iterating.

SUM is an aggregator function. It works like a measure, calculating based on the current filter context.

The following is an example of a SUMX function:

```
Total Sales SUMX = SUMX(
   'Sales OrderDetails'
   , 'Sales OrderDetails'[qty] * 'Sales OrderDetails'[unitprice]
)
```

Notice that it multiplies qty * unitprice, row by row, and then sum the results. If you added this code to a measure and put it in a visual, you would notice that the total is the expected $1,354,458.59. If we attempted to create this function using SUM, we would get a syntax error. SUM can only operate on one column. After fiddling with it, we might come up with something like this:

```
BadMeasure = (SUM('Sales OrderDetails'[unitprice]) *
Sum('Sales OrderDetails'[qty]))
```

This measure takes the total number of qty in OrderDetails and multiplies it with the total UnitPrice. This is very bad logic and results in $2,899,457,198.47. This result is very wrong.

SUMX is necessary because it preserves the row value and ties the correct qty with the correct unitprice.

Best Practice: Organize your code

This author recommends keeping measures together in one table. You may want to create more than one table for your measures. In my experience, you will end up having dozens of measures in your Power BI model. It might be difficult to find a measure if you don’t practice organization.

Best Practice: Naming Columns & Measures

When you name columns and measures, obey the following rules:

- Feel free to use spaces
- Avoid acronyms
- Make names terse, but descriptive
    - Makes Q & A easier to use
- In formulas, reference table names for calculated columns and do not reference table names for measures, so you'll know the difference

These rules will make it easier to maintain your Power BI project, data model, and reports.

Best Practice: Formatting

Power BI and DAX ignore white space. That means you can freely use whitespace to make your code easy to read and easy to navigate.

```
Days To Ship = DATEDIFF
      (
            'Sales Orders'[orderdate]
          , 'Sales Orders'[shippeddate]
          , DAY
       )
```

Figure 08-17: DAX ignores white space

DAX Expressions can have lots of parentheses and square brackets. Please use white space to control help this. Above is a good example of a properly formatted DAX expression. You can also use the website DaxFormatter.com to help you format your DAX expressions.

Other Resources

The following books can help you along your DAX journey:

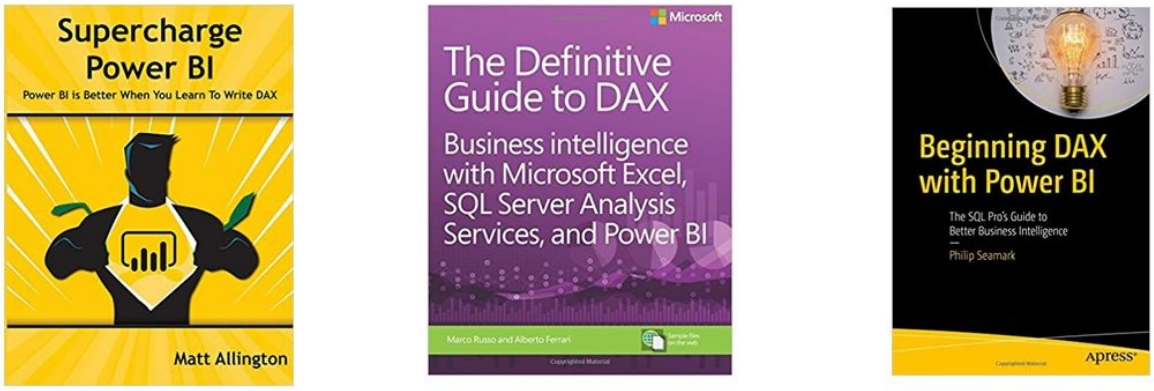


Figure 08-18: Great DAX books

DAX Studio is a great tool for learning DAX and writing DAX expressions. It was the brainchild of one of the authors of one of the books above, Darren Gosbell, Marco Russo and Alberto Ferrari. Learning this tool is essential for the budding DAX developer

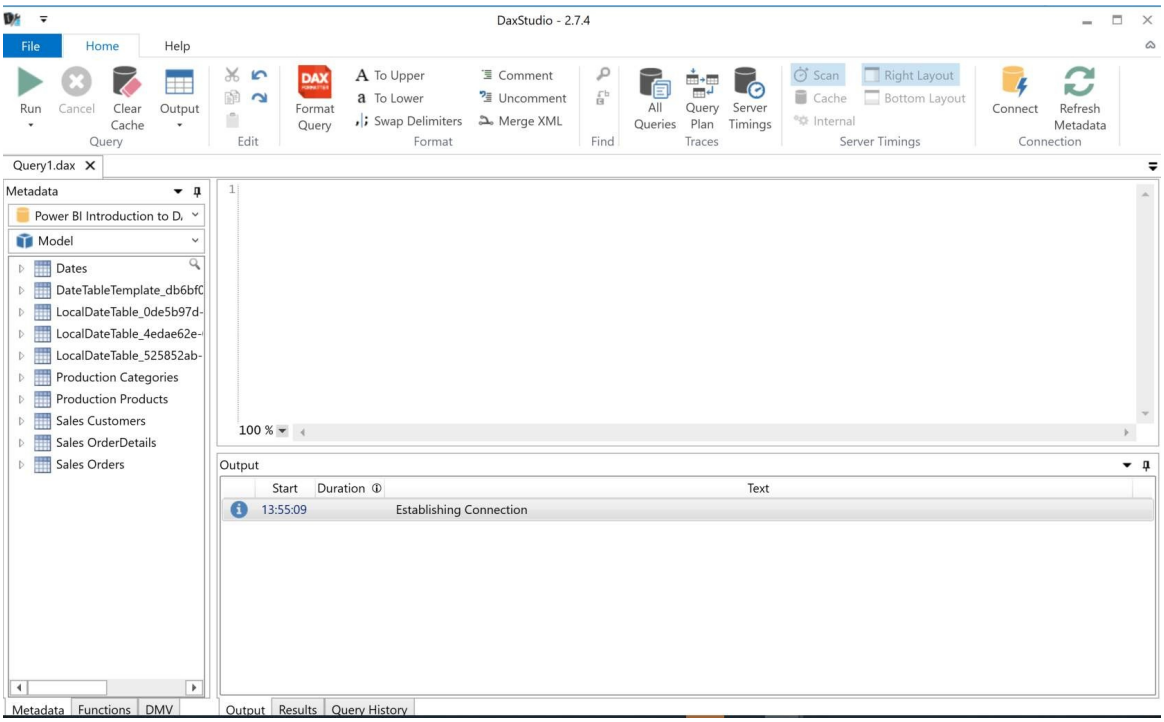


Figure 08-19: DAX Studio

Those two, affectionately called The Italians by the Power BI community, also

created a website called DAX.guide. This website, along with the DAX documentation will also accelerate your journey to DAX mastery.

Summary

This chapter helped you begin your journey to answer the job interview question "Do you know DAX?" It covered the essential elements of DAX. From here you can practice and read your way to knowing all that you need to know to answer "Yes! I know DAX really well!"

About the Author

With over 18 years of experience in databases and a current Microsoft MVP, Ike has been Microsoft certified since the beginning, currently holding an MCDBA, MCSE, MCSD, and MCT. Ike is the General Manager of Data & AI for Solliance. We have a full team of cloud data engineers and data scientists. We specialize in building highly scalable data solutions for any size of organization. Ike is a partner in Crafting Bytes, a San Diego software studio and Data Engineering group. We build software and BI solutions for companies all around the country.

In 2010, Ike founded the San Diego Tech Immersion Group (SDTIG). It has grown to be the largest user group in San Diego with over 125 active members including three other Microsoft MVPs. It is a technical book club that reads a book on a significant technical topic. Recent topics include Linux on Microsoft Azure, Angular 2/TypeScript, Data on Azure, Python for Data Scientists, and Docker/DevOps. In July 2018, SDTIG started a track on Docker/Kubernetes. We will start a new track on Databricks/Spark in November 2018. You can join virtually and watch the youtube live stream. www.sdtig.com.

Ike leads the San Diego Power BI and PowerApps user group that meets monthly. Ike is also the leader for San Diego Nerd Beers, a monthly software development social group and the co-chairman of the San Diego Software Architecture Group, a round-table of software leaders in San Diego. He also co-chairs the San Diego Software Architecture Group with Azure MVP Alumni Scott Reed.

In 2015, he co-wrote Developing Azure Solutions for Microsoft Press. He contributed all sections that related to the Microsoft Data Platform. The second edition will be released in 2018.

For more information, see http://www.ikeellis.com

Chapter 9: Power BI is Not the Same as Excel

Author: Matt Allington

Power BI and Excel are similar in what they can be used for, but they are not the same type of tool. It can be deceptive because both tools seem to do similar things using a similar approach. For example, Excel has a functional language that is used for calculations, and Power BI also has a functional language (called DAX). But that doesn't mean Excel formulas are exactly the same as DAX formulas (although they can sometimes be similar). In this chapter I will cover the differences you need to be aware of to start your Power BI journey, particularly when you come from an Excel background.

Introduction

I have been teaching people how to use Power BI, Power Pivot and Power Query since 2014 (more than 5 years at this writing). Most of the people I have taught (although not all) have come from an Excel background. That is to say that they have spent most of their business life using Microsoft Excel as their data analytics tool of choice. If you are reading this chapter, then chances are you are just the same. You love Excel but also see potential to take your reporting and analytics to the next level using Power BI.

Over my time teaching thousands of students, I have developed a deep understanding of the things that people struggle with when trying to make the transition from Excel to Power BI. I am not saying that they (or you) can't make the transition; I strongly believe that any competent Excel user can make the transition from using Excel to using Power BI. But what you must understand is that Power BI is not the same type of tool as Excel.

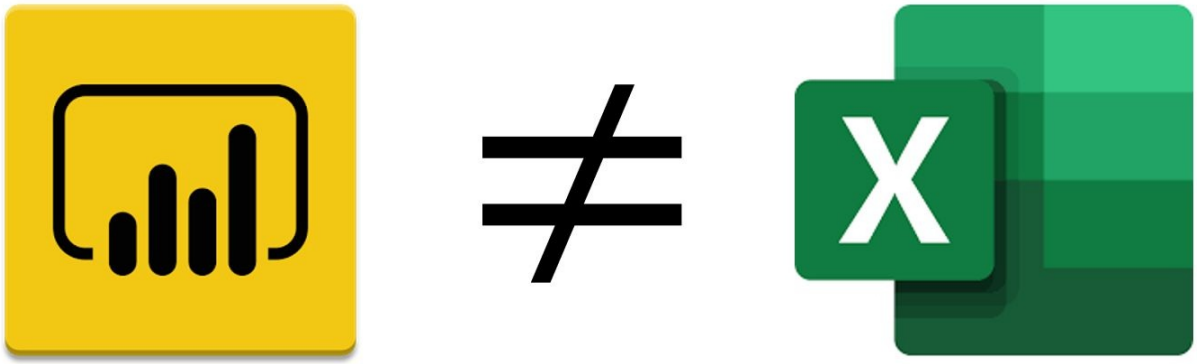


Figure 09-01: Power BI is not the same type of tool as Excel

Power BI is an umbrella term referring to a very broad ecosystem including Power BI Desktop, the Power BI Service (Powerbi.com), on premise report servers, and paginated reports to name just a subset of the total ecosystem. In this chapter I will refer to just the Power BI authoring tool called Power BI Desktop. Power BI Desktop and the PC version of Excel are the tools that are potential substitutes for each other in the world of self-service business intelligence reporting. I will be comparing Power BI Desktop to Microsoft Excel for PC, although the same comparison also applies also to Excel for Mac. At this writing, there is no Power BI Desktop for Mac.

Some Things Are the Same in Power BI and Excel

Built with You in Mind

Let me start out by talking about the similarities between Power BI Desktop and Microsoft Excel for PC. Firstly, and most importantly, Power BI Desktop was built with Excel users front of mind. Microsoft did this for a very important reason – it wanted to build a tool that was easy for Excel users to use. Microsoft corralled key people from the Excel and Analysis Services teams and set them to the task of building a new self-service BI tool. Initially the team built Power Pivot for Excel. Then a team was created to build Power Query for Excel and another team worked on Power View for Excel. Once these three "products" were underway, Microsoft had started to build out all the components needed to create the new tool that we know of today as Power BI.

DAX is a Functional Language

At the core of Power BI is the Vertipaq Engine and the DAX Language. DAX (Data Analysis Expressions) is the formula language of Power BI. Some of the older business intelligence and relational database tools do not have a functional language at all and instead have a scripted query language. Traditional SQL Server Analysis Services has a language called MDX (Multidimensional Expressions). SQL Server itself uses a scripted query language called T-SQL (Transactional-Structured Query Language). T-SQL is relatively easy to learn and MDX is quite hard to learn, but the point is that neither of these database tools have a functional language like in Excel.

A functional language is one that uses functions to perform a task. I like to think of a function like a black box – you give it some inputs and it gives you some outputs. You don't need to know "how" it comes up with the result – that is the job of the function. Consider the function SUM() in Excel. This function takes one or more parameters as input(s) and returns the addition of the values as its output. Power BI also has a functional language (DAX) and this makes it is relatively easy for Excel users to make the transition from Excel functions to DAX functions.

DAX has Many Common Functions with Excel

There are many functions that have the same or similar syntax to DAX. The example SUM() given above is a good case in point. The function has the same name and similar syntax. In Excel the SUM() function accepts ranges as inputs, and then the values in those ranges are added together. DAX is slightly different

– it accepts a single column from a single table as the input and adds the values in that column. More on this later.

Here is another example: both DAX and Excel have a function called TODAY(). In both tools, this function does not take any parameters and will return the current date from the PC.

The list of common functions is very long. You can download my DAX Quick Reference Guide PDF from http://xbi.com.au/drg

Sometimes Functions Are Similar, But Have Small Differences

There are many common functions that have the same name in both DAX and Excel but have a slightly different syntax. Take the OR() and AND() functions as 2 examples. In Excel, both of these functions can take as many input parameters as needed to solve the problem at hand.

Excel Syntax OR(logical1,[logical2],…[logicalN])

Power BI Syntax OR(logical1,logical2)

Do you spot the difference? The Excel syntax will accept 1, 2, 3, or as many parameters as you need. The Power BI syntax must have 2 and exactly 2 parameters. For this reason, I am not a big fan of the OR() and AND() syntax in Power BI. Instead I tend to use the in-line syntax as follows:

Power BI OR Inline Syntax logical1 || logical2 || logicalN

The double pipe is the “in-line” way of saying OR in DAX. The pipe symbol is the vertical bar somewhere on your keyboard. The exact location depends on the locale of your keyboard.

Power BI AND inline syntax logical1 && logical 2 && logicalN

Many Things Are Very Different Between Power BI and Excel

Although there are many similarities between Power BI and Excel, there are many more things that are very different. The rest of this chapter is dedicated to explaining these differences so that you understand them and know how to approach Power BI differently to Excel.

Power BI is a Database, Excel is a Spreadsheet

Let me start off by defining what I mean by a database and a spreadsheet.

Database

A database is a tool that is built to efficiently store and retrieve data. There are different types of databases including transactional databases (like SQL Server) and reporting databases (like SQL Server Analysis Services SSAS). Power BI is more like SSAS than it is like SQL Server.

The Power BI database is a reporting database. It is not designed to allow you to edit or alter the data that is loaded, but instead to faithfully and efficiently produce summary reports of the actual data loaded.

Spreadsheet

A spreadsheet is a 2-dimensional page that contains multiple cells in rows and columns. It is an unconstrained "canvas like" space not unlike a canvas used by an artist to create a picture. One of the greatest strengths of a spreadsheet is that you can do anything you want with the space. One of the greatest weakness of a spreadsheet is, you guessed it, you can do anything you want with the space. A spreadsheet is a doubled edged sword. The freedom and flexibility to do what ever you want is its greatest strength and its greatest weakness.

A Spreadsheet Used as a Database

While not wanting to confuse matters any more than needed, the simple fact is that many business people tend to use Excel as a database even though it is actually a spreadsheet. I am not saying it is wrong to do this, it is just that Excel lacks the checks and balances that exist in a database tool to prevent accidental errors being created. Have you ever wondered why a 500,000 row spreadsheet can take 20 mins to update when you make a change? This is why – it is a spreadsheet and was not designed for its primary purpose to be a database.

One of the key things that Microsoft has set out to change with Power BI is to provide a more robust reporting database tool designed to be used and maintained by business users. Actually, Power BI is also an enterprise strength BI tool. Power BI is designed to serve both self-service and enterprise BI users.

Differences Between a Spreadsheet and a Reporting Database

There are quite a few differences between these things; here are just a few of them.

Excel	**Power BI**

Can change one or more data points by just typing over the top.	The data must be changed prior to (or during) load. Once the data is loaded, you can't change it.
Data needs to be in a single table before you can summarize the data using a Pivot Table or Pivot Chart.	Data doesn't need to be consolidated into a single table prior to summarizing using a Matrix or any other visual. In fact you are encouraged to keep the data in separate tables as a star schema.
Excel formulas are created and then copied. Each cell therefore has its own unique copy of a formula (except array formulas of course). It is possible that formulas across cells can get out of sync causing potential errors.	DAX formulas (measures) are written once. The same measure is then used over and over again in different visuals. There is only one single definition of the measure, so there can be no exceptions and discrepancies. The results of each calculation are not permanently stored but only used when needed to be displayed in a visual.
	DAX formulas (calculated columns) can be written in the data view to extend a loaded table of data. This looks very similar to Excel, but you should choose to use measures instead of calculated columns where possible (so the data is not permanently stored).
Each copy of the formula can be edited as needed. You can write formulas like SUMIF() to create on the fly filters and summaries.	A single measure is re-used with different "filters" applied to get different results from the same measure.

Figure 09-02: Differences between a spreadsheet and a reporting database

Tips to Get You Started as You Move to Power BI

The rest of this chapter will focus on my key tips to help get you started on your Power BI journey. What you do with this information is important, and it is related to your teachability index. The Teachability Index is defined as:

```
Ability to Learn × Willingness to Change
```

There is no point learning something new if you are not willing to change. There is no point being willing to change if you don't know what needs to be done differently. What is your teachability index?

DAX Calculated Columns vs Measures vs Tables

Both Power BI and Excel have a functional language that a business user can use to extract insights from the underlying data. But unlike Excel, the DAX language can be used in three very different ways within Power BI Desktop:

- DAX Measures
- DAX Calculated Columns
- DAX Tables

Each of these approaches to writing DAX are valid, and they all use the same DAX language. All approaches have their place and purpose in Power BI, however in my experience, most beginners make some simple mistakes and select the wrong approach.

Excel Users Write Too Many Calculated Columns

The most common problem I see Excel users make is to write too many calculated columns. I am not saying it is wrong to write a calculated column – what I am saying is that most Excel users tend to default to writing calculated columns when they would be better off writing measures instead.

There are a few reasons why Excel users tend to default to writing calculated columns rather than measures.

- New Power BI users are often self-taught, and they don't know the difference between a calculated column and a measure. Indeed, often they don't even know that measures even exist.
- When you switch to the data view in Power BI, it looks a lot like an Excel spreadsheet. Excel users feel very comfortable when adding a new column to a table that looks like this, because it feels like home.

ExtendedAmount	TotalProductCost	TaxAmt	Freight	RegionMonthID	DiscountSell
564.99	308.2179	45.1992	14.1248	Australia6	451.99
564.99	308.2179	45.1992	14.1248	Australia6	463.29
24.49	9.1593	1.9592	0.6123	Australia6	23.02
8.99	3.3623	0.7192	0.2248	Australia6	7.82
8.99	3.3623	0.7192	0.2248	Australia6	7.19
564.99	308.2179	45.1992	14.1248	Australia6	474.59
53.99	41.5723	4.3192	1.3498	Australia6	40.49
53.99	41.5723	4.3192	1.3498	Australia6	42.65
120	44.88	9.6	3	Australia6	102.00
7.95	2.9733	0.636	0.1988	Australia6	6.60
24.49	9.1593	1.9592	0.6123	Australia6	20.33

Figure 09-03: Data view in Power BI looks like an Excel spreadsheet

- The DAX syntax is normally easier in a calculated column, and it is less common to get strange error messages that you don't understand (as is often the case when writing measures for the first time).

SQL Users Write Too Many DAX Table Queries

The most common problem I see SQL Server professionals make is to write too many table queries. This is understandable I guess, because SQL professionals are comfortable writing queries and DAX is used as a query language when creating tables. But as the saying goes, "just because you can, doesn't mean you should". There is a time and a place for DAX queries and DAX tables, but it is often not the best approach in Power BI (sometimes, but not often).

Everyone Should Be Writing Lots of Measures

The new shiny thing that sets Power BI apart from Excel is the introduction of Measures. Measures are not a new concept – they have been available in SQL Server Analysis Services for many years. But measures are new to most Excel users, and they are not always the first thing that comes to mind to SQL professionals. Unless someone tells you that you should be using more measures, how would you even know? I'm telling you now – so now you know.

Measures have a lot of benefits over Calculated Columns and Tables. I like to think of the data you load into Power BI being a bit like raw cooking ingredients, and the measures you write being like a "recipe" to "cook" something up using the raw data. The recipe is repeatable – you can use it over and over again against the same ingredients to get the same result.

Benefits of measures include:

- Measures do not create duplications of your data. Conversely, both calculated columns and tables duplicate the data in Power BI.

- You write a measure once, and then use it many times.
- Each measure that references one or more columns can be given its own unique name making it easy for end users to find and use the business insights they need.

g.:
tal Sales = SUM(Sales[Value])
g Transaction Value = AVERAGE(Sales[Value])

te how the above 2 measures reference the same column, but they deliver a different business insight. The name clearly communicates what the measure is providing.

- Each measure has its own bespoke formatting that is fit for purpose. e. g. even though Total Sales and Avg Transaction Value above are calculated from the same column of data, both of these measures can have their own data format

g.:
tal Sales = $29.36 M
g Transaction Value = $486.09

- Measures can use data from multiple columns and tables within the same formula. This cannot be done with a simple drag and drop of a column into a visual in Power BI.

Using Visuals to Structure your Output

When you first open a new report in Power BI, you are faced with a daunting sight. The Power BI Report canvas looks a lot more like Power Point than it does Excel.

Figure 09-04: PowerPoint's, Power BI's and Excel's canvas

Power BI intentionally has a blank reporting canvas. When you want to visualize your data, you need to add one or more visualisations to the reporting canvas to create the layout structure for your report.

Creating Structure in Excel

Excel has a 2 x 2 grid to store your data. You can put any bespoke number, text,

formula etc, in each and every cell on the sheet. There is little or no constraint on how you must proceed to structure your report. It is possible that you may end up with a nice, orderly, tabular layout in your Excel spreadsheet, but of course it may not end up that way if you are not careful.

Creating Structure in T-SQL

Writing queries using T-SQL uses a different approach to a spreadsheet. You write code using the T-SQL query language. The structure of the tabular output and all the aggregations are configured inside the T-SQL script itself (see the example below).

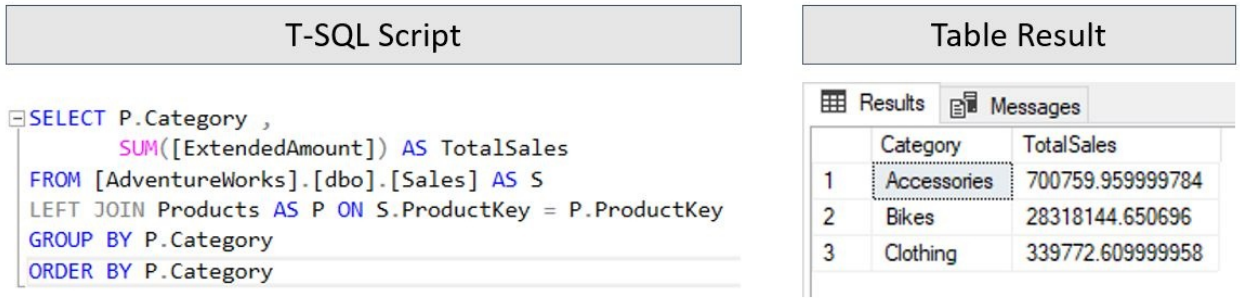


Figure 09-05: Creating structure in T-SQL

Creating Structure with DAX Queries

When you write DAX Queries in Power BI to produce a table, the process is quite similar to T-SQL above. This is why many SQL professionals like to write DAX queries. Here is a DAX query that could be considered equivalent to the T-SQL Script above.

```
EVALUATE
SUMMARIZECOLUMNS (
    Products[Category],
    "TotalSales", SUM ( Sales[ExtendedAmount] )
)
ORDER BY Products[Category] ASC
```

Figure 09-06: Creating structure with DAX queries

Despite being able to write DAX queries as shown above, Power BI is not designed to use this approach as the primary way to create summary data. Instead, you should prefer to use Power BI visuals to create the structure.

Creating Structure in Power BI with Visuals

Both the Excel approach and T-SQL approach outlined above are very different to the Power BI approach. When using Power BI, the structure of the output is first selected from one of the available visuals.

Each visual will implicitly shape and control the structure of the final output. If you don't like the result, it is very easy to change it to a different visual (try doing that in Excel).

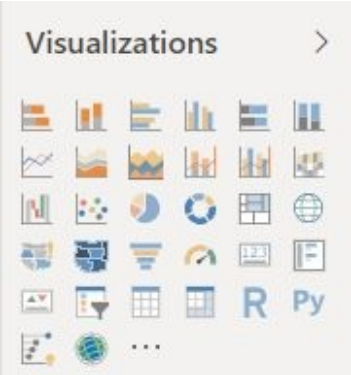


Figure 09-07: Standard visuals in Power BI

I always teach Excel users to start with a Matrix, as this is the closest match to an Excel pivot table and a sheet. Using a Matrix as the starting visual is a good way to get used to using Power BI. Another good thing about using the Matrix visual is you get to "see" the numbers. This makes the experience more "Excel like", and it is an important visual confirmation that you are doing something right.

After adding the visual to the canvas (as can be seen in 1 below), data can be added to the matrix against Rows (2), Columns (3) and Values (4). This process is very similar to how you would build a pivot table in Excel.

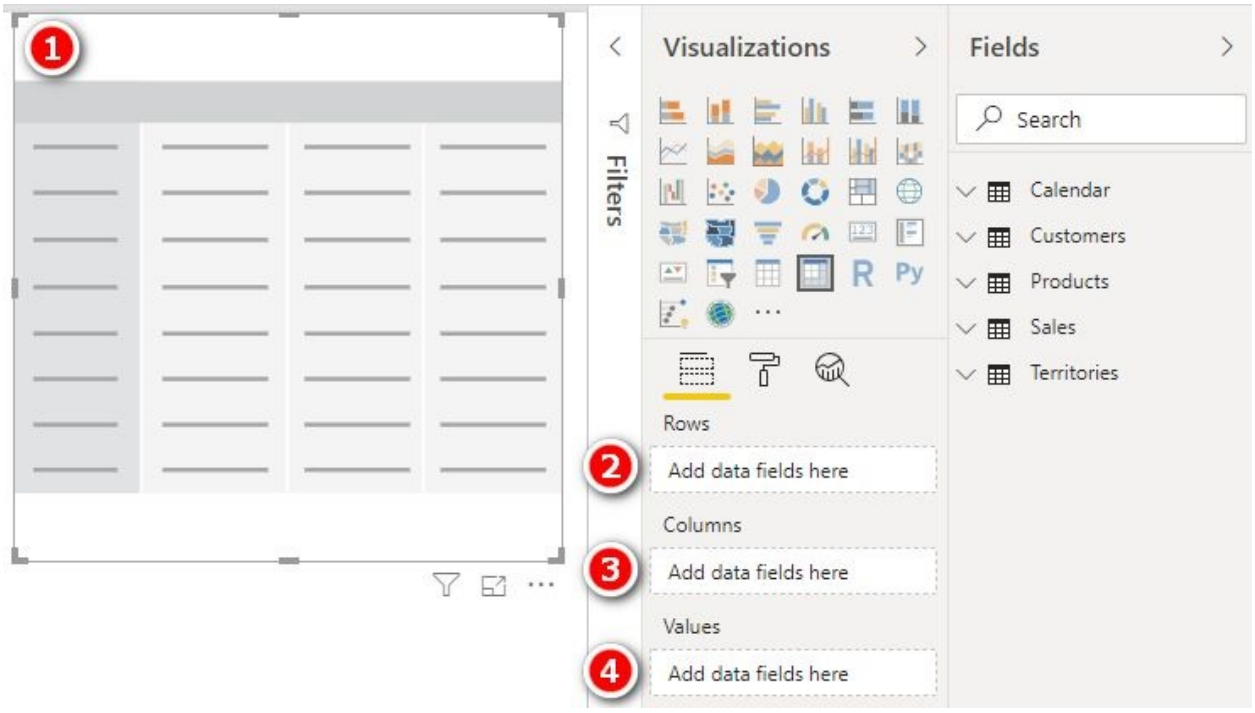


Figure 09-08: Steps to create a visual in Power BI

In the image below, I have built the same table created earlier using T-SQL but I have used the Matrix visual to create the structure of the result. The big difference is that the table below was built without writing a single line of code. I simply added the matrix visual, dragged the Product[Category] column into the Rows and the Sales[ExtendedAmount] column into the Values – Power BI did the rest.

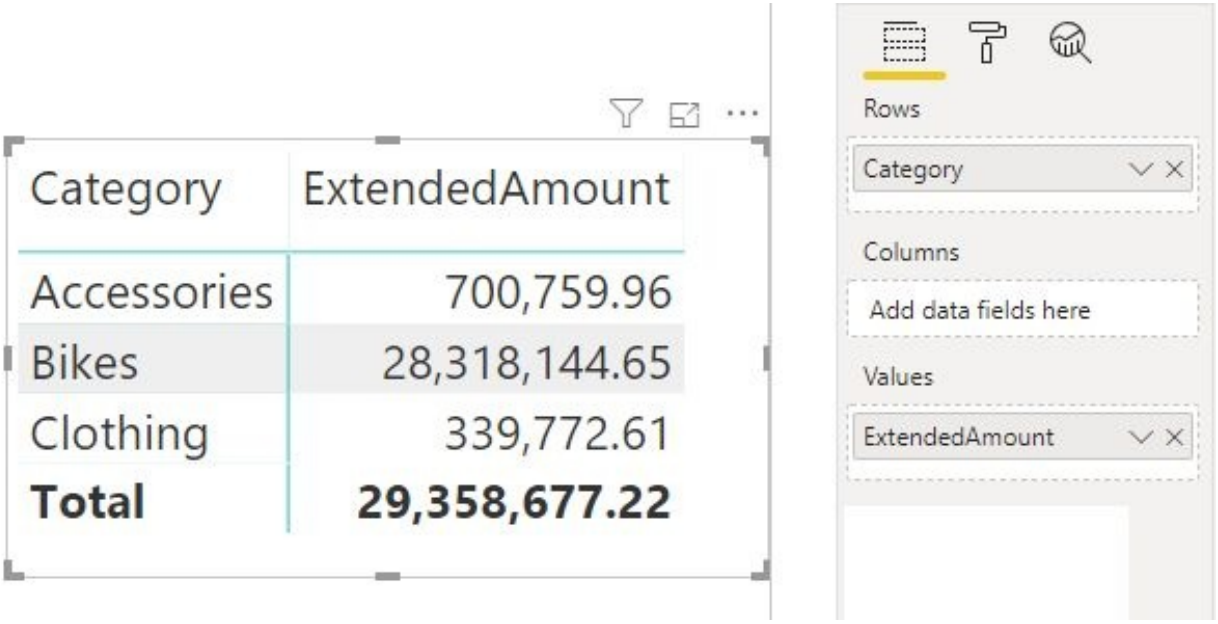


Figure 09-09: A Matrix visual with columns Category and ExtendedAmount

Note: Sales[ExtendedAmount] is a column of numbers, not a measure. When I added the column to the Values above, Power BI was smart enough to add the numbers up for me without writing a measure.

Why Write Measures at All, Then?

OK, so now you are asking why bother to write measures at all when you can just drag a column of numbers into a visual. Well, the short answer is that you don't have to write measures if you don't need to. The basic concepts such as adding up numbers in a column, finding the average in a column, etc can all be achieved by dragging a column of data to the Values section of the visual, and then setting the aggregation behaviour for that data in the column. If you don't like the default aggregation for the column, you can change it by selecting the drop-down menu (shown in 1 below) and then selecting a different aggregation from the available options (2 below).

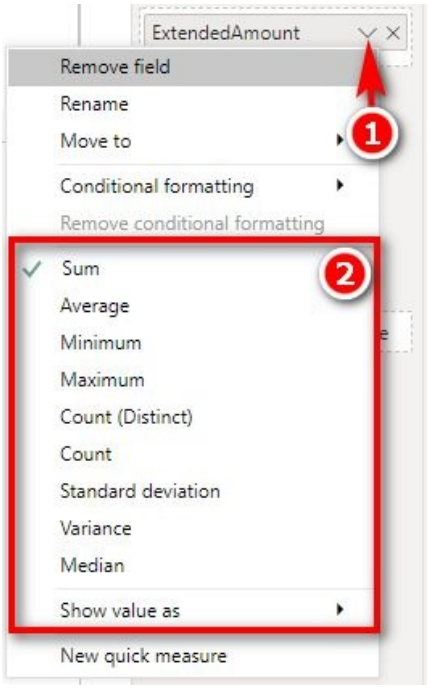
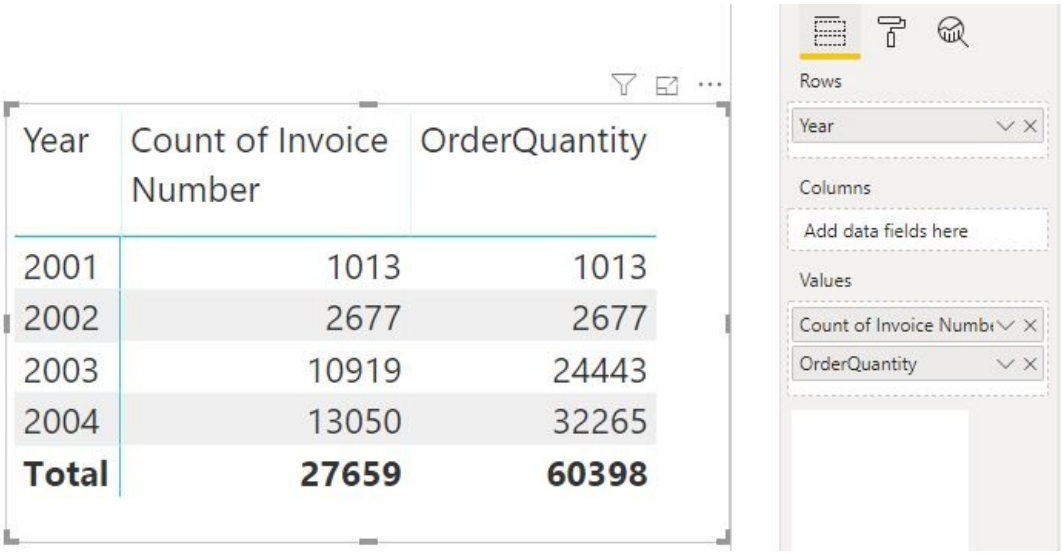


Figure 09-10: How to select the aggregation for column ExtendedAmount

But while the above approach will work, you will very quickly hit the limits of capability. Take the following example:

- You can drag a column to work out the total number of invoices
- You can drag a column to work out the total number of items sold



Year	Count of Invoice Number	OrderQuantity
2001	1013	1013
2002	2677	2677
2003	10919	24443
2004	13050	32265
Total	**27659**	**60398**

Figure 09-11: A matrix visual over Year, Invoice Number, and Order Quantity

But you can't drag a column to find out the average number of items per invoice. You need data from two columns to complete this calculation, and for that you will need to write a DAX Measure.

I have written the DAX measure to calculate the average number of line items per invoice, and you can see it in the formula bar in the image below.

```
1 Avg Line Items per Invoice = DIVIDE(sum(Sales[OrderQuantity]) , DISTINCTCOUNT(Sales[Invoice Number]))
```

Year	Count of Invoice Number	OrderQuantity	Avg Line Items per Invoice
2001	1013	1013	1.00
2002	2677	2677	1.00
2003	10919	24443	2.24
2004	13050	32265	2.47
Total	**27659**	**60398**	**2.18**

Figure 09-12: A matrix visual by Year, Invoice Number, Order Qty, and Avg Line Items per Invoice

In addition to needing a DAX measure when you are referencing multiple columns of data, don't forget all the benefits of having better business names for

your measures and also being able to apply suitable formatting for each individual business concept. Compare the table above with the DAX version below and note the changes/improvements in the names and the formatting.

Figure 09-13: A matrix visual over Total Line Items, Total Invoices, and Avg Line Items per Invoice

```
Total Line Items = SUM(Sales[OrderQuantity])

Total Invoices = DISTINCTCOUNT(Sales[Invoice Number])

Avg Line Items per Invoice = DIVIDE([Total Line Items] , [Total Invoices])
```

Figure 09-14: DAX formulas for Total Line Items, Total Invoices, and Avg Line Items per Invoice

I hope you agree the good business names, the suitable formatting, and the reuse of measures inside other measures all make the small extra effort of writing measures worthwhile. Believe me, writing measures and learning to write DAX is the secret sauce that will set the Power BI superheros apart from the rest of the pack.

Filter First, Calculate Second

In Excel, you can write bespoke, one off formulas in any cell you like. That is not how it works in Power BI. In Power BI, you first take ALL the data you are given in one or more columns or tables. You write generic formulas (Measures) using those column and table inputs and use the visuals to filter the data before returning the results.

Use the Visuals to Filter the Data Before Returning the Results

I want you to read that heading again "use the visuals to filter the data before returning results". This is one of the most important concepts you must understand about Power BI. Let me refer back to the Matrix I created earlier (shown again below for convenience).

Year	Total Invoices	Total Line Items	Avg Line Items per Invoice
2001	1,013	1,013	1.0
2002	2,677	2,677	1.0
2003	10,919	24,443	2.2
2004	13,050	32,265	2.5
Total	**27,659**	**60,398**	**2.2**

Figure 09-15: A matrix visual over Total Line Items, Total Invoices, and Avg Line Items per Invoice

As you can see in the visual above, there are 5 different numbers in the Matrix for [Total Invoices] despite the fact that I only wrote one DAX Measure. The DAX measures is as follows:

Total Invoices = DISTINCTCOUNT(Sales[Invoice Number])

Despite only writing a single formula, you can see I have 5 different results. How can that be? The answer is that the **Visuals in Power BI filter your data** – that's what they do.

I am not suggesting you would be surprised to see 5 different numbers for Total Invoices in the Matrix above. What I am saying is that you need to understand HOW those 5 different numbers were created.

The process is as follows.

- The Row section of the visual is displaying the Calendar[Year] column.
- The visual takes the first year (2001 in this case) and filters the Calendar table.
- The filter on the Calendar table flows through the relationship in the direction of the arrow (shown in 1 below in the Model view) onto the sales table (no VLOOKUP required here).

Figure 09-16: Flow of filters from one table to another

- Now the sales table is filtered for all sales in 2001 as a result of the filter flowing from the Calendar table to the sales table. We say that the filter propagates from the Calendar table to the Sales table.
- The DAX formula then is calculated for just those sales that remain after the filter is applied to all the tables.
- Every row, column, bar, line, pie slice, total, sub-total etc in Power BI are ALL calculated this way.

All DAX formulas are always evaluated using this approach. Filter first, evaluate the result second after all filters have been applied. If you remember nothing else from this chapter, this is it.

Filter First, Evaluate Second

Understanding filter propagation is essential to writing good DAX.

About the Author

Matt Allington is a Power BI Consultant, Trainer and MVP specialising in helping business users learn and apply Power BI to solve business problems. Matt has over 35 years' commercial and IT experience using data to get things done. Matt is the author of the bestselling book "Supercharge Power BI – Power BI is Better When You Learn to Write DAX" http://xbi.com.au/scpbib and he runs regular live online training courses teaching people how to use Power BI http://xbi.com.au/scpbi

Part IV: AI and Power BI

Chapter 10: AI for Business Users in Dataflow and Power BI Desktop

Author: Leila Etaati

In this chapter, an overview of how to use AI in Power BI service and Power BI desktop will be discussed. For business users, always using AI is about easy access to the tools without writing any codes and activate AI capability with a couple of clicks. In this chapter two different possibility of consuming AI, will be discussed. First how as a business user we are able to analyse the text in Power BI service will be shown. In the next part, how using some AI powered visuals such as Key Influencer in Power BI desktop to analyse the data without knowing the machine learning concepts.

Cognitive Service in Power BI

Cognitive Services are an asset of APIs, SDKs and services that can be use by developers to add AI to their applications. Cognitive services in Azure has five major categories as Language, Vision, Speech, Search and Decision. Each of those has a selection of different services. These services have been used by many developers in web, and mobile applications. Moreover, there is a way to use them in Power BI reports [1].

One of the interesting services is about language analysis or Text Analytics. Text analytics can range from detecting the main keywords in a sentence, identify how much a customer is happy, detecting the main entity and structure of a sentence, detect the language of a sentence and so forth.

As a developer in Power BI Desktop, there is a way to use the Text Analytics API in Power Query [2]. However, this process is a bit hard for Business users or people who are not familiar with Power Query. In this section, First, how a business user is able to use cognitive services for the aim of text analytics and extract the main keywords of the text.

AI in Dataflow

There is an announcement about the availability of using Cognitive Services in Power BI Service, Dataflow. Dataflow is the new feature in Power BI service that allows people to transform their data in the cloud and do the ETL in the Power BI service.

To access the AI in Dataflow you need to have a premium account.

To start, you need to login to your Power BI Service account, and you need to create a new workspace (not “My Workspace”). You will not be able to create Dataflow in “My workspace” (Figure 10-01).

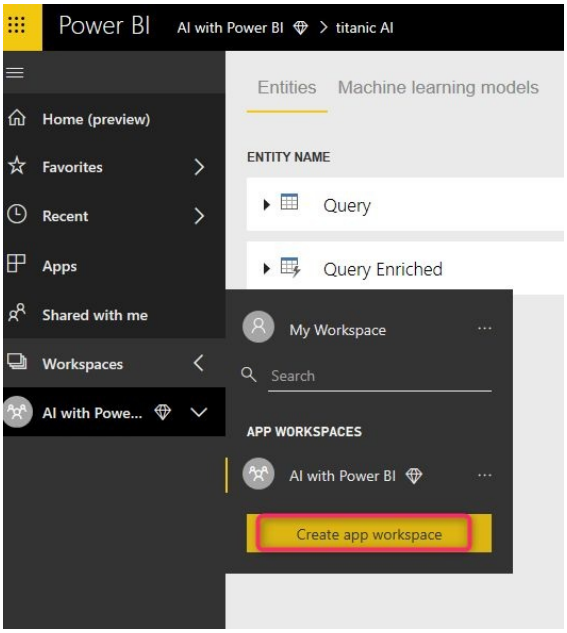


Figure 10-01: Create New Workspace in Power BI Service

To create a new workspace, click on the “Workspaces”, then click on the “Create app workspace”. Now, in the new page, you are able to create a new workspace (pro or premium). Now you can create a new one by putting the name, description and image for the workspace. So, choose the proper image, name, and description and click on the Save (Figure 10-02).

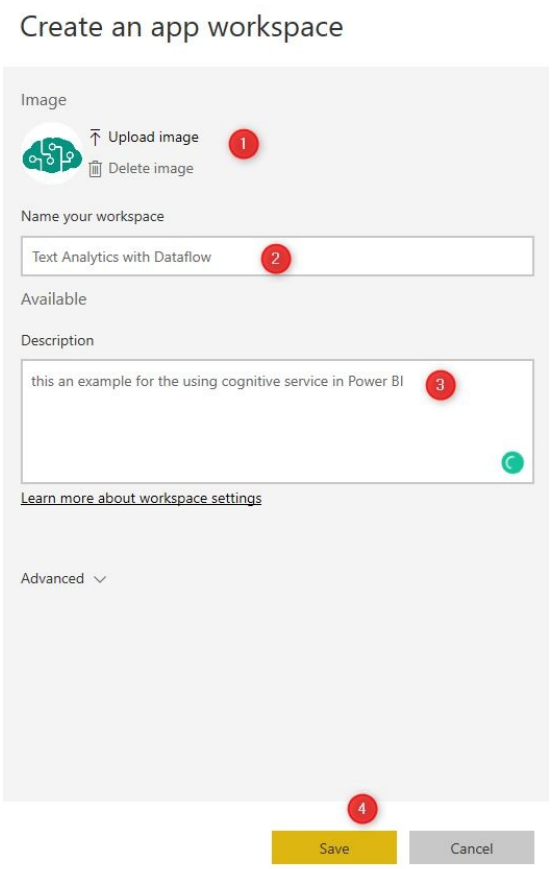


Figure 10-02: Create New Workspace by providing some Details

In the new page, you can see your new workspace and the welcome page, just skip the welcome page to navigate to the main workspace.

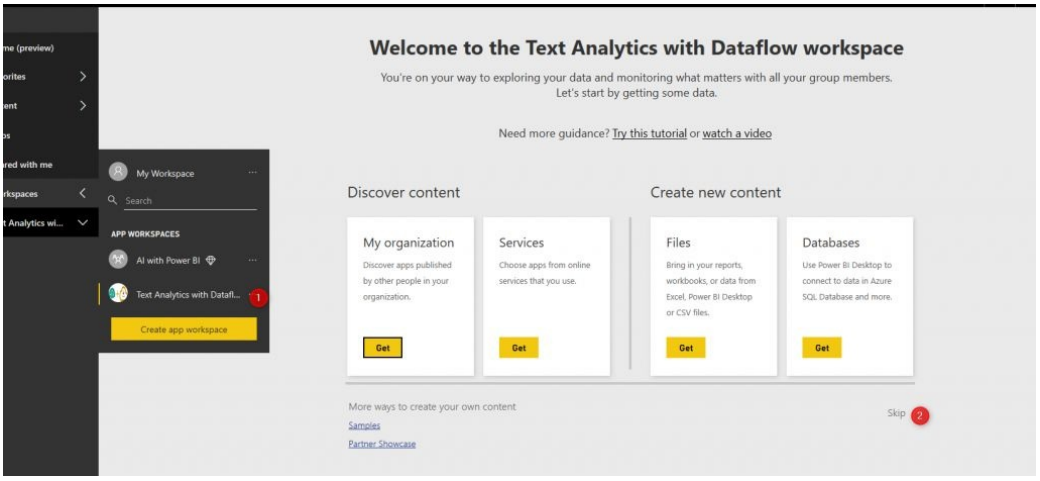


Figure 10-03: Workspace Welcome Page

In the main workspace page, beside Dashboard, Reports, Workbooks, and Datasets, we a have new tab named Dataflow. Click on Create option at the top

right of the page and choose the Dataflow.

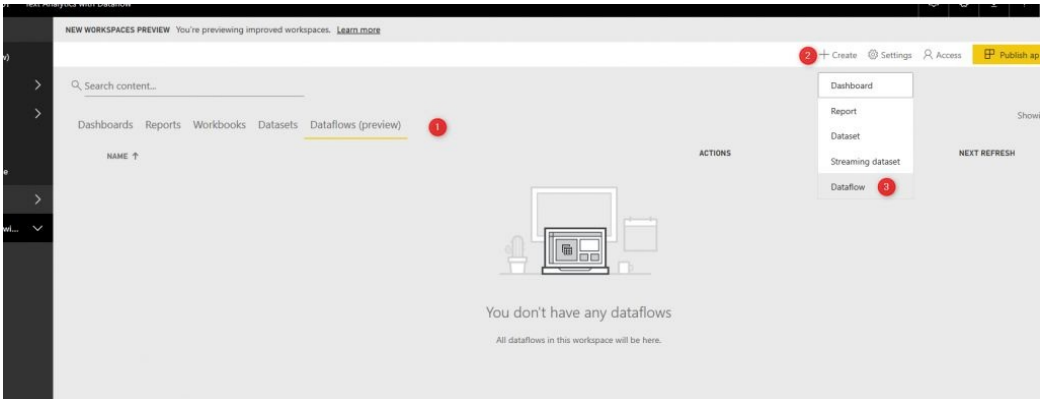


Figure 10-04: Create New Dataflow in Power BI Service

By creating a new Dataflow, Power BI Service navigates you to a new page that is asks you about adding entities (data source) click on the “Define new entities” (Figure 10-05).

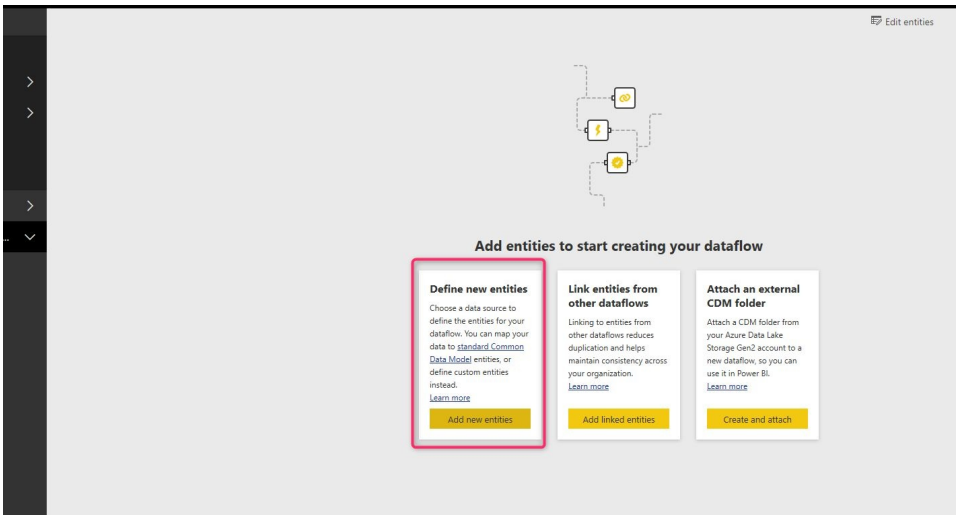


Figure 10-05: Define New entity page

In the next page, you can connect to different resources from cloud to on-premises. For this example, I just put some comments people put for some products, to access the data navigate to the files folder, data for chapter 17. Click on the Blank Table.

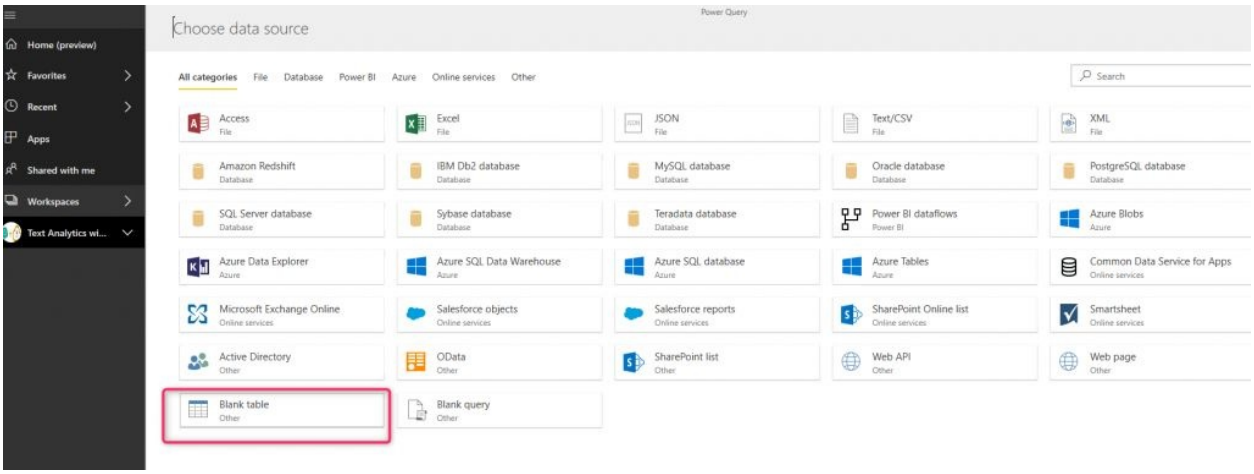


Figure 10-06: Create a Blank Table

Then change the title of the column to Comments, put the text for each row (row by row). then put a name for the table and click on the Next.

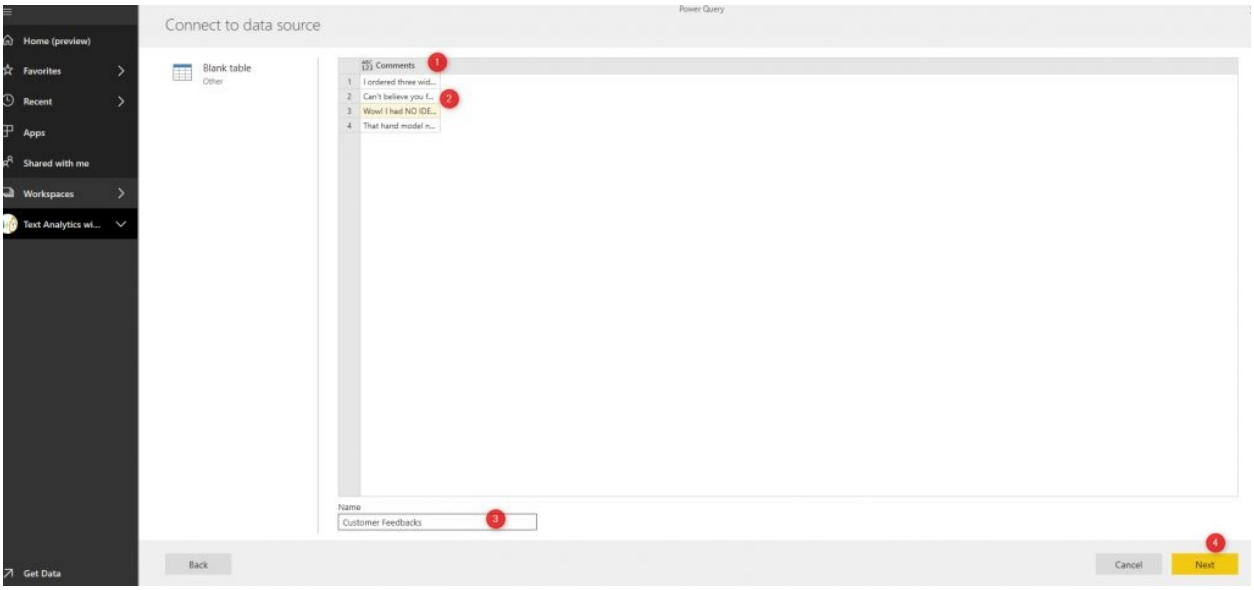


Figure 10-07: Create a Blank Table and Put the text

The new query will be generated and navigate into a new page as Power Query editor in Power BI web service. As you can see in the below Figure 10-08, It has some of the features of the Power Query Editor from transforming the columns, to combine and so forth. For the aim of text analytics, there is an icon on the top of the page as AI Insight, click on it.

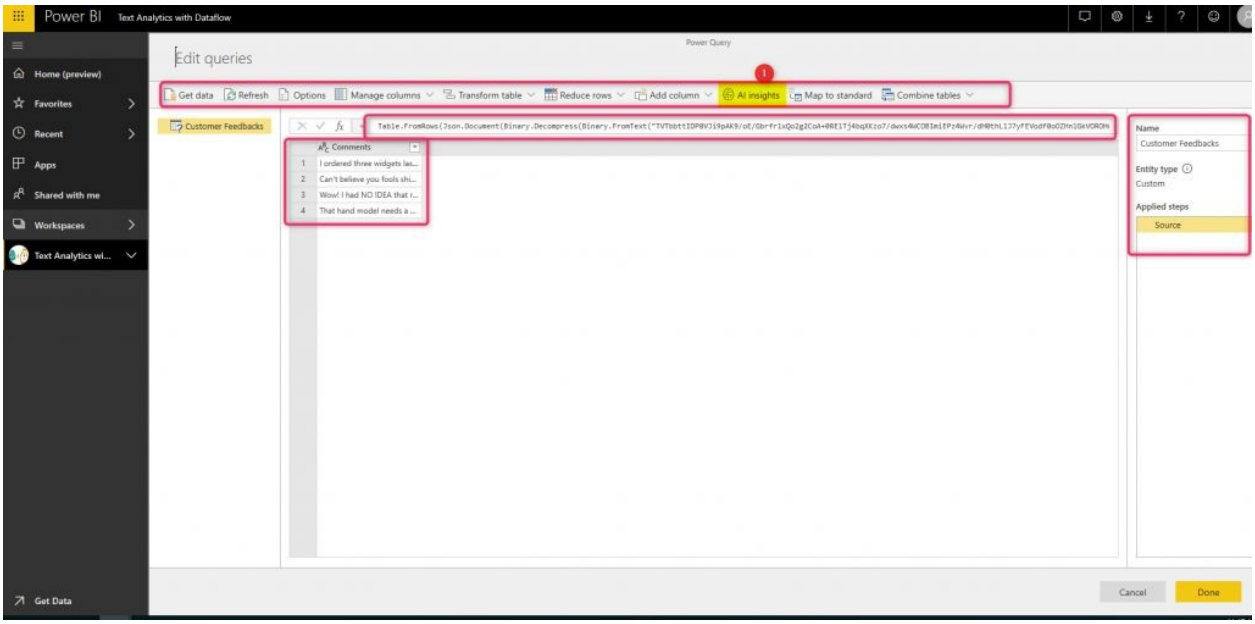


Figure 10-08: Edit Data in Dataflow

Then a new page will show up that you will see the cognitive service folder (if you already create one for Azure ML there should be a folder for Azure ML as well). Expand the cognitive services folder and you will be able to see four different cognitive services

- Sentiment Analysis
- Keyword Extraction
- Language Detection and
- Image Tag

Click on the last one that is *Sentiment Analysis*

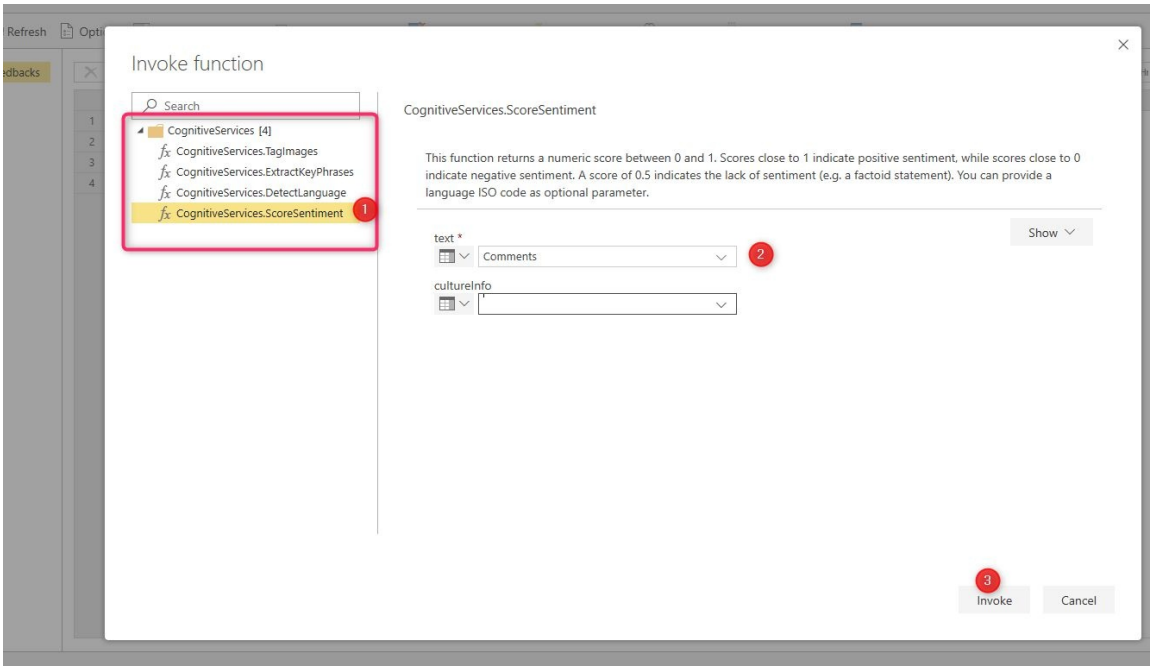


Figure 10-09: Cognitive Service in Dataflow

Sentiment Analysis is for identifying how much a customer is happy with products. The value will range from 0 to 1.

If the value is close to 1that means the customer is happy about the product, otherwise if the value is close to 0 that means the customer is not happy about the product.

Choose the sentiment analysis service, it will aske about the target column as *Text*, then you need to choose the column you want to apply this service. In this example the *Comment* should be chosen as the target column.

Then, just click on *Invoke*. A new page will be shown that shows the original column (Comment) plus a newly added column name *Cognitive Service*. This new column has a value from 0 to 1 for each row.

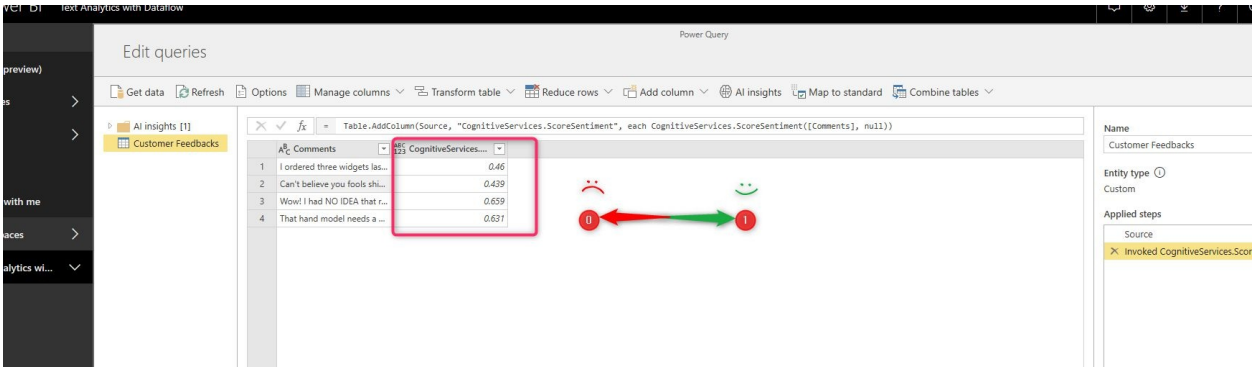


Figure 10-10: Invoke Sentiment Analysis in Power BI Service

Now we got the result, we need to save the result so we can use it in Power BI Service report or Power BI Desktop. Click on “done” at the bottom of the page to apply all changes. Then, you need to save the query by clicking on the top right of the page and put a name for the query.

Then, under the Dataflow you can see your new Dataflow. You just need to click on the Refresh Button. In this example, first I am going to get the data in Power BI desktop.

To do that, open your Power BI Desktop, and sign in with the account you used for the Dataflow. Click on Get Data, choose the Power BI dataflow, you my need to sing into your power BI service that you created the dataflow.

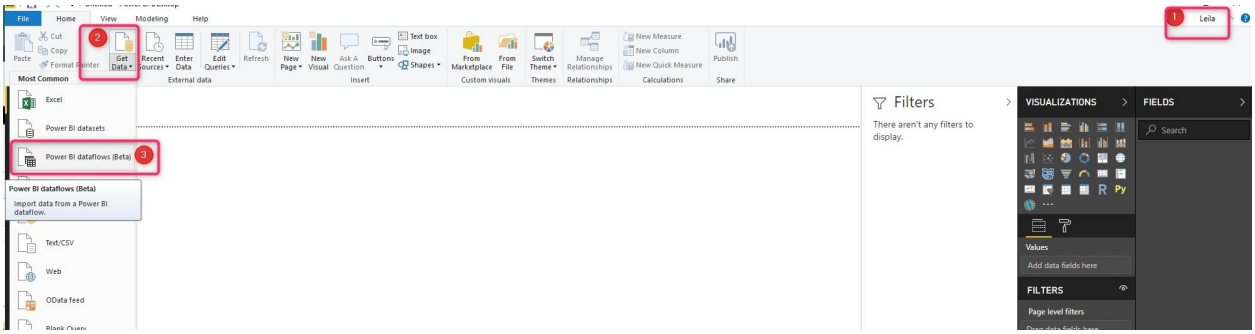


Figure 10-11: Connect to Dataflow from Power BI Desktop

After signing in, now you can see the list of Dataflow you already created. Expand the list and see an overview of the query with Comments and sentiment score columns. Now by loading the data you are able to create a report in Power BI Desktop.

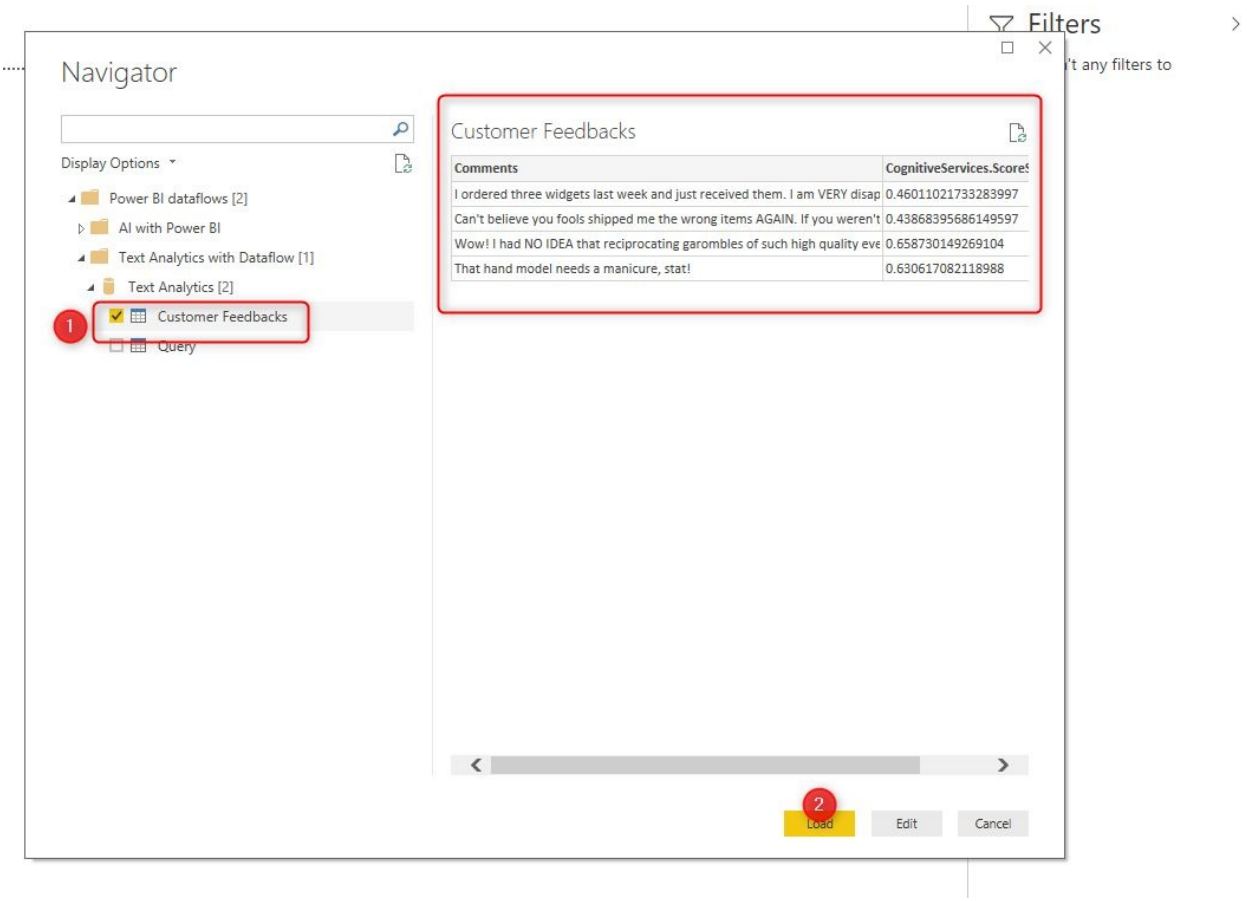


Figure 10-12: Dataflow in Power BI Desktop

You can also extract the Keywork and the language of the text following the same process.

However, for the image tagging you may need to follow some more steps. In the next section, I will explain the process.

Image Tag in Power BI

Image tag is another service in Cognitive Service that allows to extract the object in the image and for each detected object, it provides the confidence level.

To see a demo on this service you need to navigate to the https://azure.microsoft.com/en-us/services/cognitive-services/computer-vision/#analyze

Then just import a picture and see the result. For example, I uploaded a picture of my dog and I got the below results.

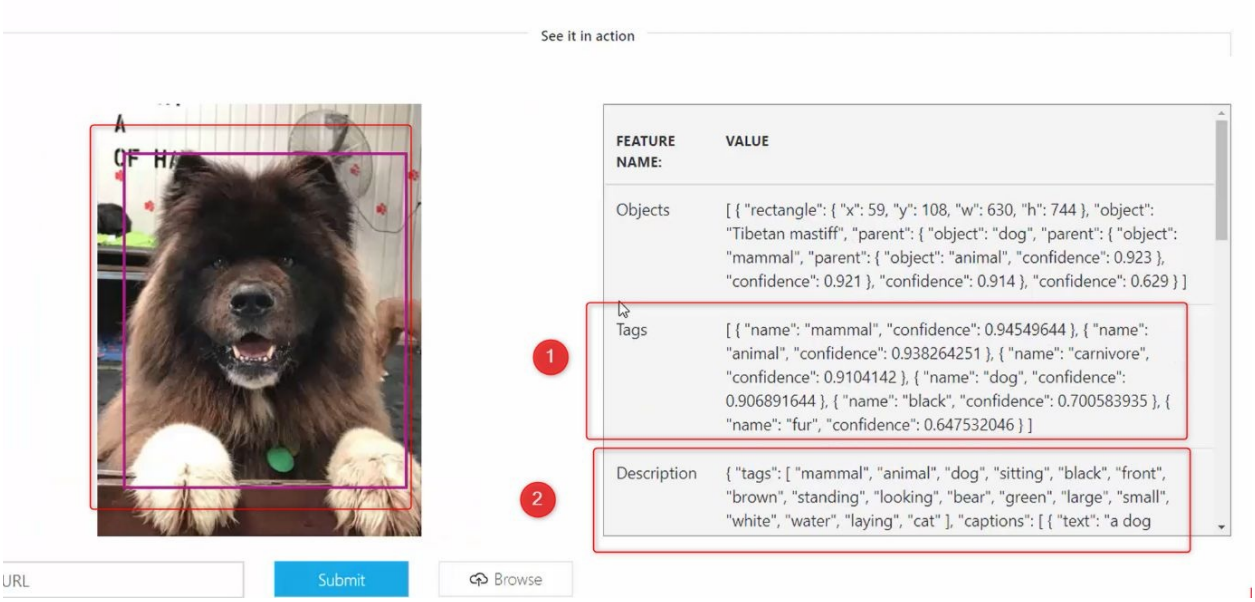


Figure 10-13: Image Tag in Cognitive Service

This service is available in Dataflow alongside with other language service. I already uploaded some of the pictures into my Azure Blob storage and in this section, I will show how to apply image tag on them using Cognitive Service function available in AI Insight.

I have some links to my pictures, also I need to write some code to extract them from the blob storage (Figure 10-14).

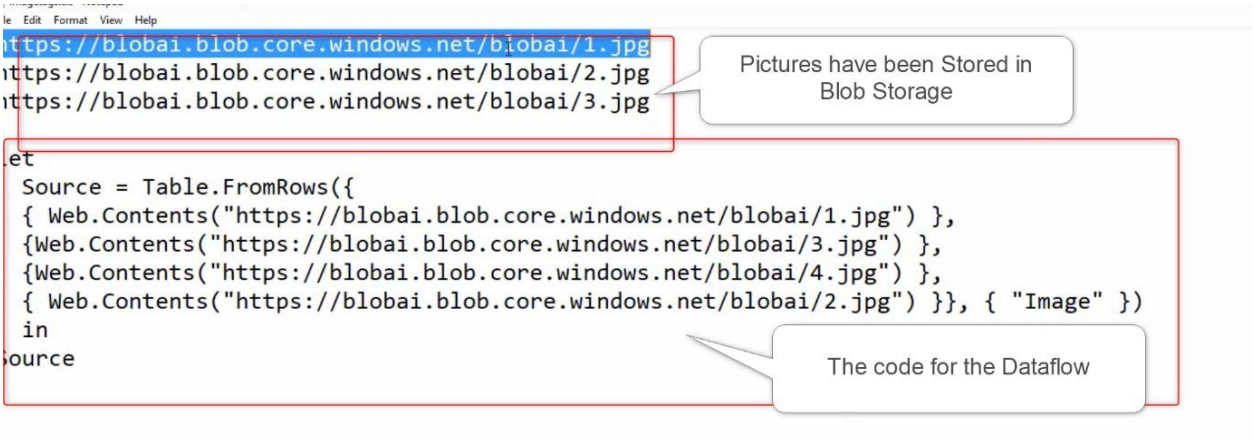


Figure 10-14: Image Links in Blob Storage and the Required Code to extract them.

Next, get data in Dataflow (same as the previous section), this time instead of choosing the get data from text, choose the get data from Blank Query.

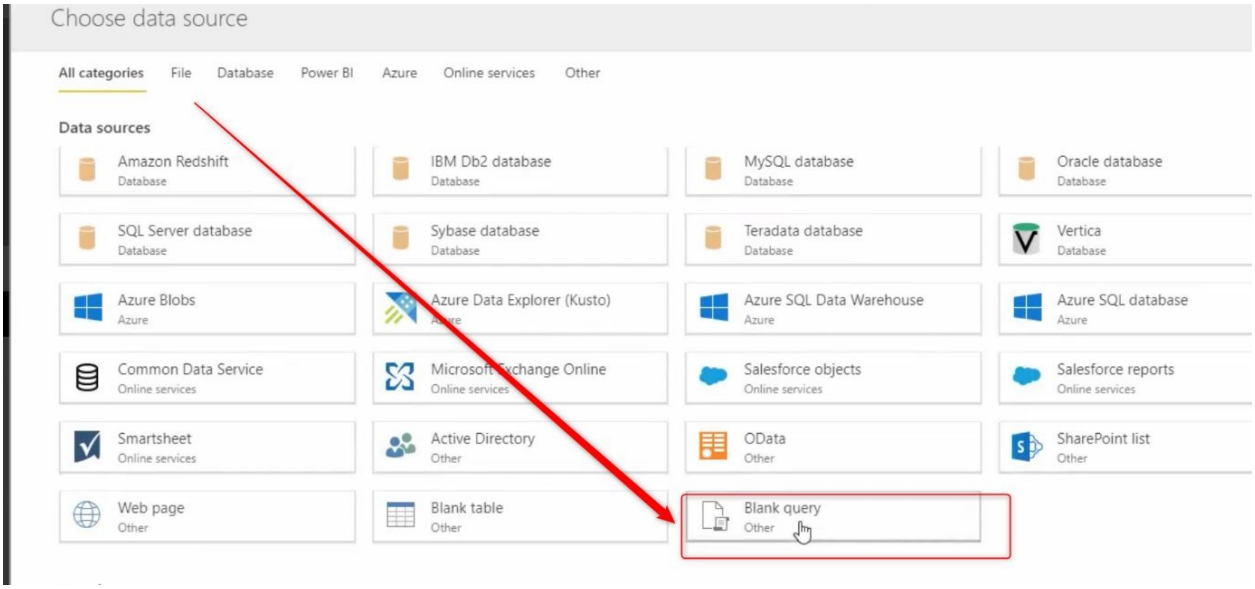


Figure 10-15: Get Data from Blank Query

In the new page delete the existing code and replace it with below code (you should have your own image link).

let

Source = Table.FromRows({ { Web.Contents("image address 1") }, {Web.Contents("image address 2") }, {Web.Contents("image address 3") }, { Web.Contents("image address 4") }}, { "Image" })

in Source

Figure 10-16: Get Data Using Blank Query

After loading the data, you should see the imported image as a column with binary format.

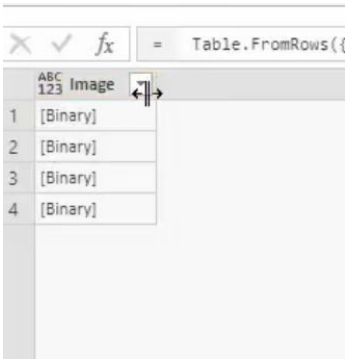


Figure 10-17: Loaded Image with Binary Format

Next, click on the AI Insight at the top of the page, and this time choose the Image Tag function.

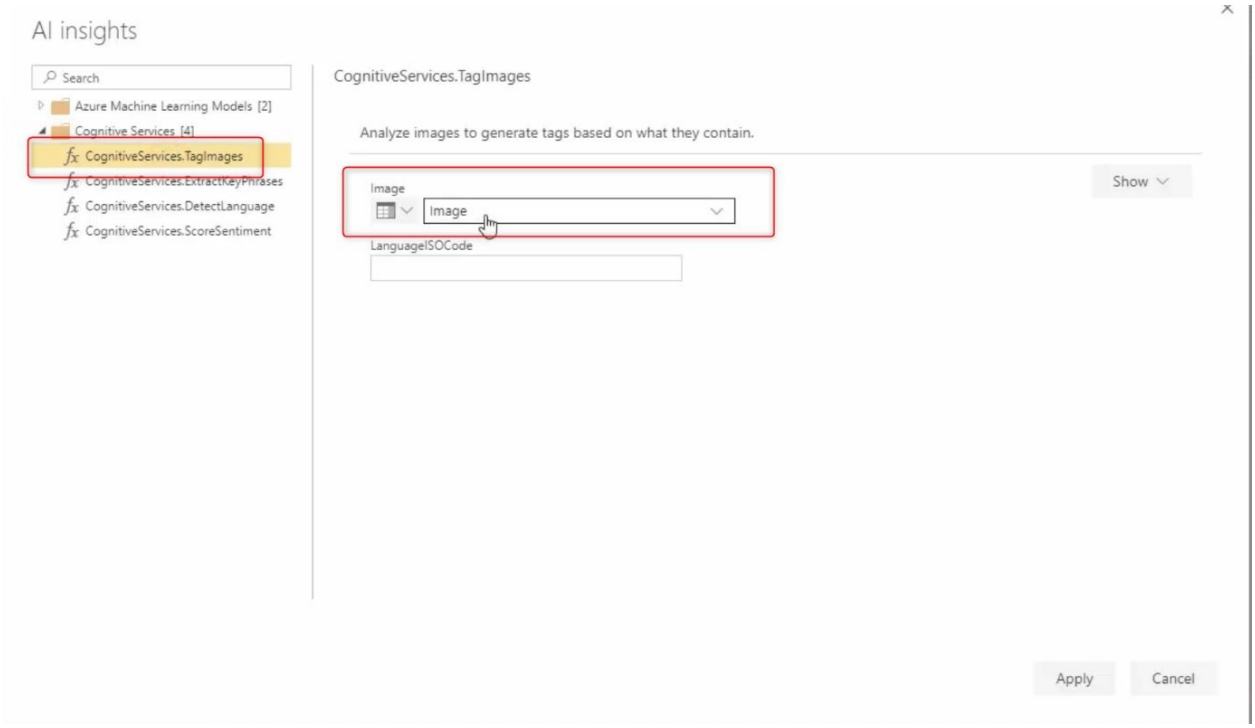


Figure 10-18: Image Tag Function

Select the Image column and click on “Apply” to see the results. Then you need to expand the result to see two columns: one is the image tag and another one has more details such as image tags and related confidence.

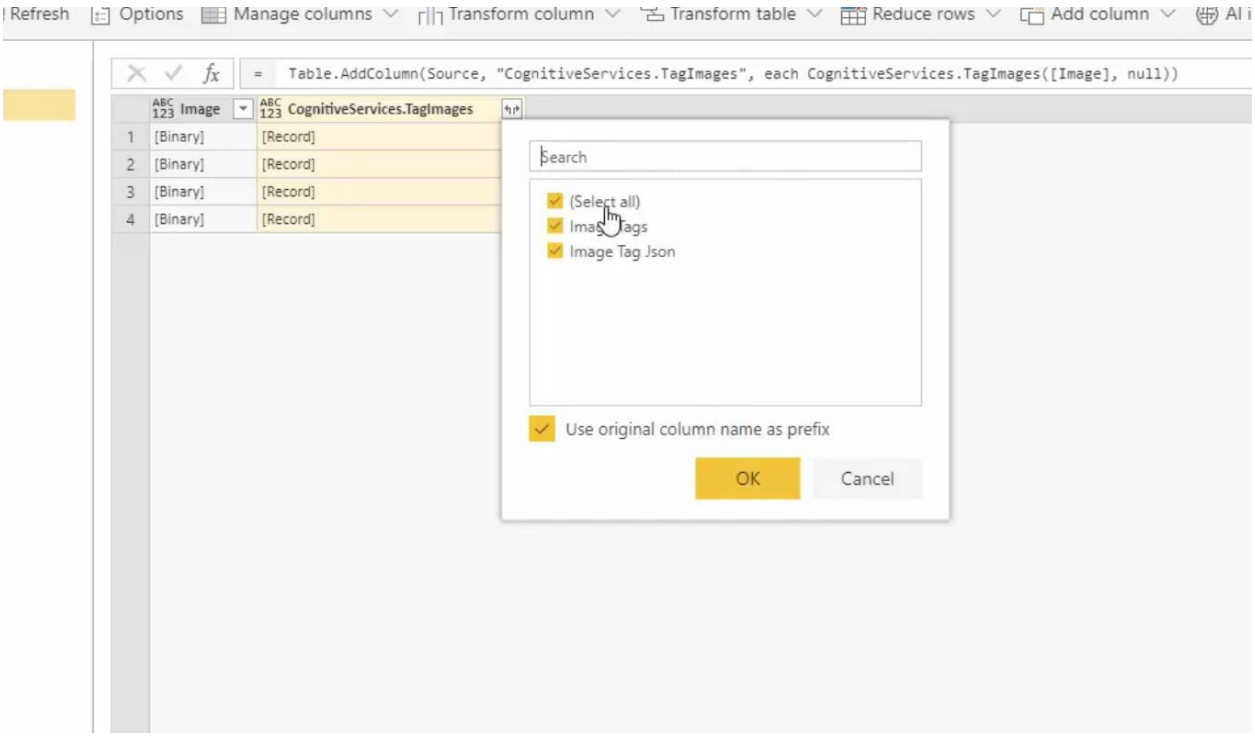


Figure 10-19: Image Tag Results in Dataflow

Just expand the results to see the tags and confidence. Same as text analytics, you can see the query in Power BI Desktop. You need to put a name and refresh the dataset in Dataflow, then open your Power BI Desktop and from “get data” choose the Dataflow. The result is like a JSON format, so you need to do some data transformation to see the confidence, but the Image tag column is already cleated.

Figure 10-20: Image Tag Results in Power BI Desktop

Key Influencer

In this section, I am going to demonstrate a new visualization that has been released by the AI team in Power BI in recent months. Before showing off this nice feature, there are some key points about this visual.

- It can be used by Data scientist, Data Engineer and End users
- It is easy enough to use and interpret
- It consumes lots of algorithms behind the scene to identify the main factor
- It can be used to align with other customer visuals to create a better visualization

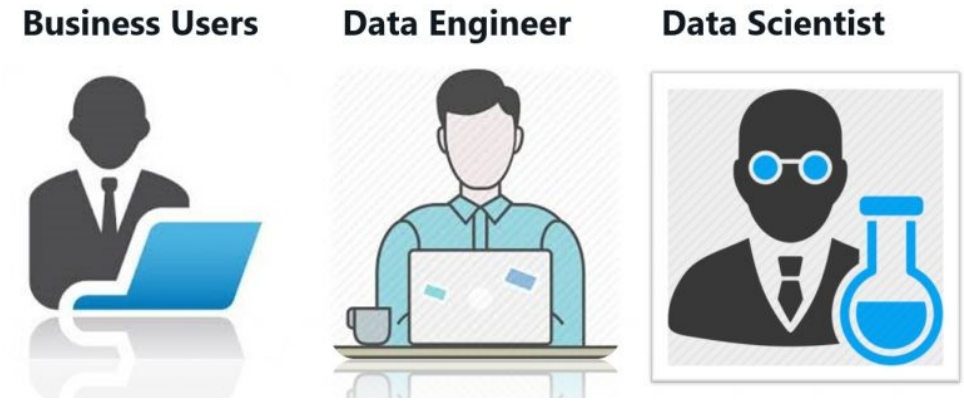


Figure 10-21: Who can use Key Influencers Visual

Moreover, this visual:

- Interpret both Categorical and Numeric variables
- Provides a great clustering approach: find the natural grouping on data, and then visualize the main top cluster (segment) and also, it shows how a combination of factors affects the metric that you're analysing.
- Explain the results: the visual provides a brief description of how it works [1]
- As mentioned before, this visual employs a combination of algorithms. In categorial and regression analysis different algorithms are used.

Figure 10-22: Analysing categorial and Numeric Values

In this section, I am going to use a dataset about concrete. Concrete is used in building the bridge, buildings and so forth.

The main elements for creating concrete is

- Cement: A cement is a binder, a substance used for construction that sets, hardens, and adheres to other materials to bind them together.
- Blast Furnace Slag: stony waste matter separated from metals during the smelting or refining of ore.
- Fly Ash: Fly ash or flue ash, also known as pulverized fuel ash in the United Kingdom, is a coal combustion product that is composed of the particulates that are driven out of coal-fired boilers together with the flue gases.
- Water: The amount of water in concrete controls many fresh and hardened properties in concrete including workability, compressive strengths, permeability and water tightness, durability and weathering, drying shrinkage and potential for cracking [2].
- Superplasticizer: Superplasticizers, also known as high range water reducers, are chemical admixtures used where well-dispersed particle suspension is required. These polymers are used as dispersants to avoid particle segregation and to improve the flow characteristics of suspensions such as in concrete applications [3].
- Coarse Aggregate: Coarse aggregate is the portion of the concrete which is made up of the larger stones embedded in the mix. Concrete contains three ingredients; Water, cement, and aggregate. That aggregate is made of fine sand and coarse gravel.

- Fine Aggregate
- Age: how many days

The dataset available from here: http://archive.ics.uci.edu/ml/machine-learning-databases/concrete/compressive/Concrete_Data.xls

So, let's start to predict what will be the strength of the concrete regarding other elements such as ashes, water, and so forth.

List of Questions

I want to answer the following questions:

- What factors have more impact on the strength of the concrete to decrease or increase and how much?
- I am interested in seeing the natural classification of my data.
- Also interested in seeing some rules like if the amount of Cement is … and age is … then what is the strength

Let's Answer these three questions using a brand-new visualization named **Key Influencers.**

Get it!

The key influencer is a preview feature, to access it you need to follow below steps

Click on File –> Option and Settings –>Options –> then click under Global, click Preview Feature, and you should find the Key Influencer Visual at the bottom

You need to restart the Power BI (close and open again)

Just notice, it is a preview feature, some enhancement will be applied on it soon

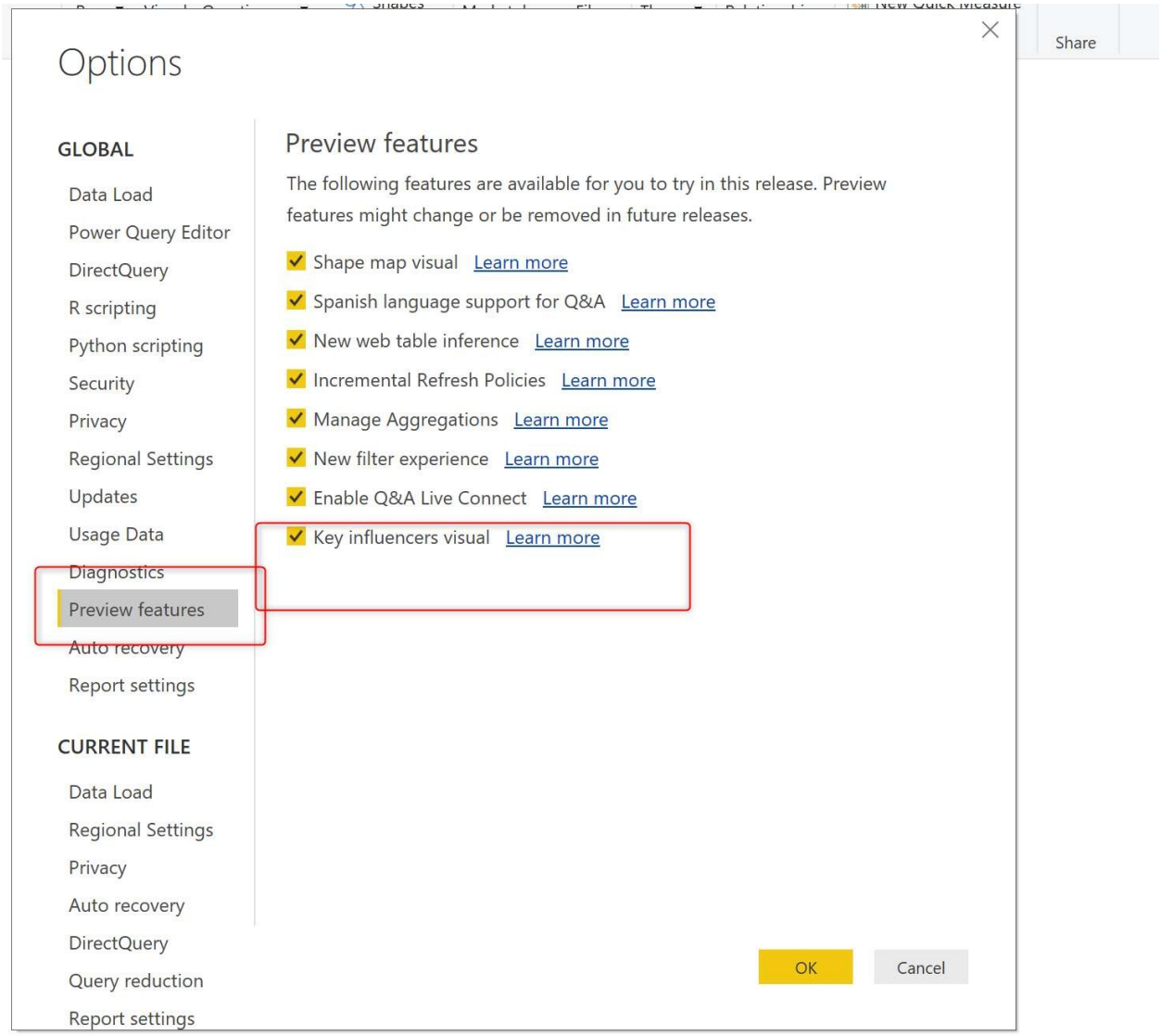


Figure 10-23: Enable Key Influencers Visual

Use It!

Now you need to import the concrete dataset into Power BI Desktop

Get Data –> CSV –> Load

Now Our plan is to analyse the strength of the concrete, hence click on visual that has been added to the Visualization panel, and for the analyse choose **Strength** field from **Concrete** dataset. You can see in the Figure 10-24.

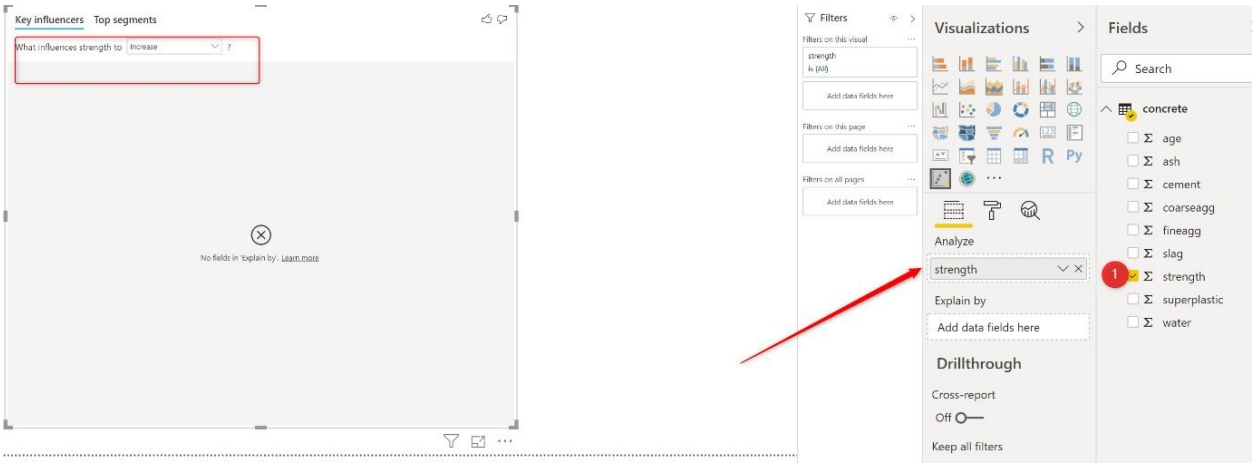


Figure 10-24: Load Data and Select which Column to Analyse

Now, you drop other columns into “Explained By” data field and see the visual show some analytics about what will impact on the strength of the concrete.

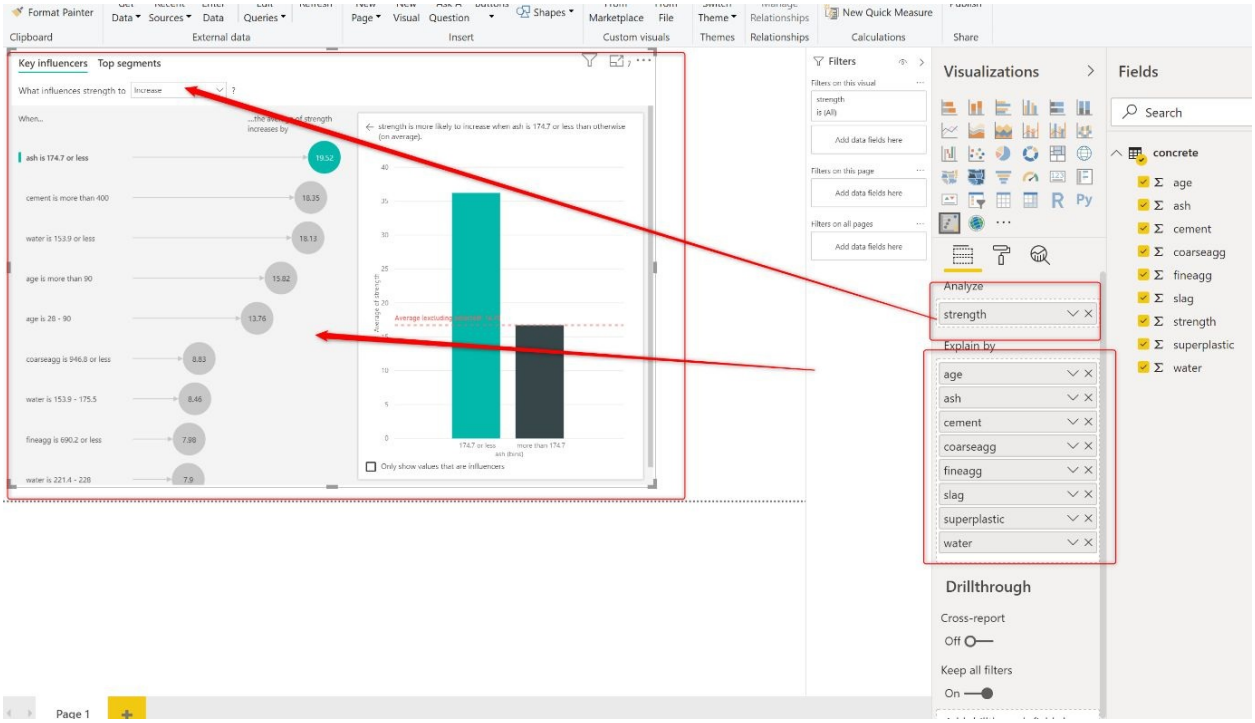


Figure 10-25: Fill out the Explained by Data Fields

The first important point, the list of the factors are displayed in the left panel with specific order. The one at the top has most impact on the strength of concrete than others. As you can see in the Figure 10-26, If the cement increases 400 then the strength will increase by 18.35 in another word it impacts by 4.5%.

Figure 10-26: Concrete Strength Analysis by Key Influencers Visual

There is another analysis tab named Top Segments. Click on the Top segment at the top and choose the first segment with 61.12. As you can see in the Figure 10-27, in Segment 1, the average strength is 61.12 and the average age of cement is more than 3 days and amount of cement is between 159.1 and 262.

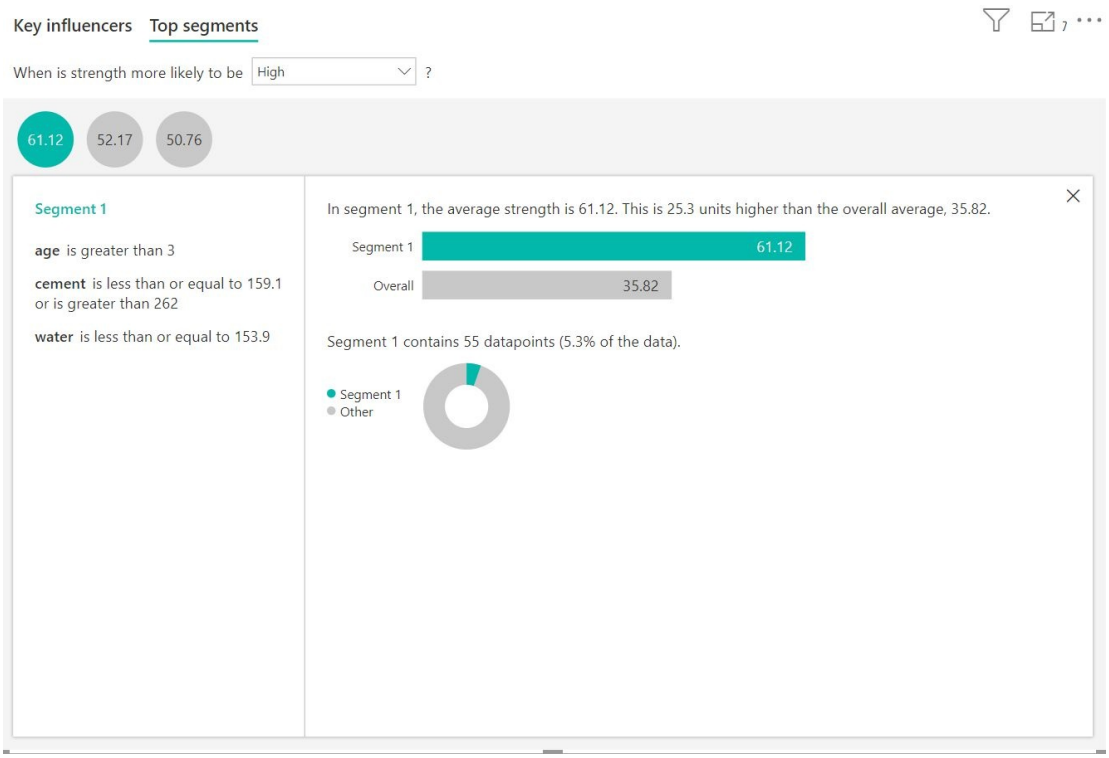


Figure 10-27: Top Segment Analysis

This visual uses the decision tree algorithms to extract the rules about which attribute will impact on the strength of cement. As a business user, you are not running any algorithm, and by using the Key Influencers visual the analysis will be automatically shown to you. This example was about a continues value (strength of cement) but you can also use it for the categorial variable analysis as well.

Summary

In this chapter, some of the features for AI that can be used by the business users without writing any code in Power BI service and Desktop have been explained. First, an explanation on how to apply text analytics on the customer feedback was discussed. How users can see the customer feedback in a numeric scale has been shown. Next, how we to do image tagging has been presented. Finally, the analytical Key Influencers visual that provides more insight out of data has been presented.

All these features presented in this chapter can be done by Business Users, Developers and Data Scientists.

About the Author

Leila is the first Microsoft AI MVP in New Zealand and Australia. She has a PhD in Information System from the University of Auckland. She is the Co-director and data scientist in RADACAD Company with more than 100 clients around the world. She is the co-organizer of Microsoft Business Intelligence and Power BI Use group (meetup) in Auckland with more than 1200 members, She is the co-organizer of three main conferences in Auckland: SQL Saturday Auckland (2015 till now), Difinity (2017 till now) and Global AI Bootcamp 2-18. She is a Data Scientist, BI Consultant, Trainer and Speaker. She is a well-known International Speakers to many conferences such as Microsoft Ignite, SQL PASS, Data Platform Summit, SQL Saturday, Power BI World Tour and so forth in Europe, USA, Asia, Australia and New Zealand. She has over ten years' experience working with databases and software systems. She was involved in many large-scale projects for big-sized companies. Leila is an active Technical Microsoft AI blogger for RADACAD.

Chapter 11: AI in Power BI Desktop

Author: Markus Ehrenmüller-Jensen

AI is everywhere – and now even included in Power BI Desktop. Sometimes AI might be very apparent when you enrich your data with predictions by explicitly calling an Azure Machine Learning web service in Power Query. Sometimes it might by hidden in an analytic feature offered by Power BI's interface.

No matter if you are a business user, analyst or data scientist – Power BI has AI capabilities tailored to you. This chapter will cover how you can integrate and leverage the use of languages DAX and R, and how to integrate Azure Cognitive Services and Azure Machine Learning Services to make more out of your data.

Introduction

Microsoft is currently investing heavily in making Artificial Intelligence (AI) available for everybody. After Self-service BI we are now facing Self-service AI. In this chapter we will concentrate on the capabilities of Power BI Desktop (not the Power BI Service, as in the previous chapter). While the possibilities are almost endless, I will concentrate on two AI functionalities, linear regression and text mining, and walk you through the steps necessary to implement both with different features within Power BI Desktop: Visuals & Analytics, DAX, R, Azure Cognitive Services, and Azure Machine Learning Services. While the chapter's limitations in term of pages do not allow for a comprehensive introduction to neither Linear Regression and Text Mining, nor to DAX, R, Azure Cognitive Services and Azure Machine Learning Services, it gives you an impression of what steps you need to take and of how to get started. Take the examples as templates from which you can derive solutions for problems of your own organisation.

Linear Regression

Linear regression is a quite simple mathematical formula to create e. g. a trend line for a measure over time. In our example I will use simple linear regression for sales amount over order date, like the straight blue line in this screenshot:

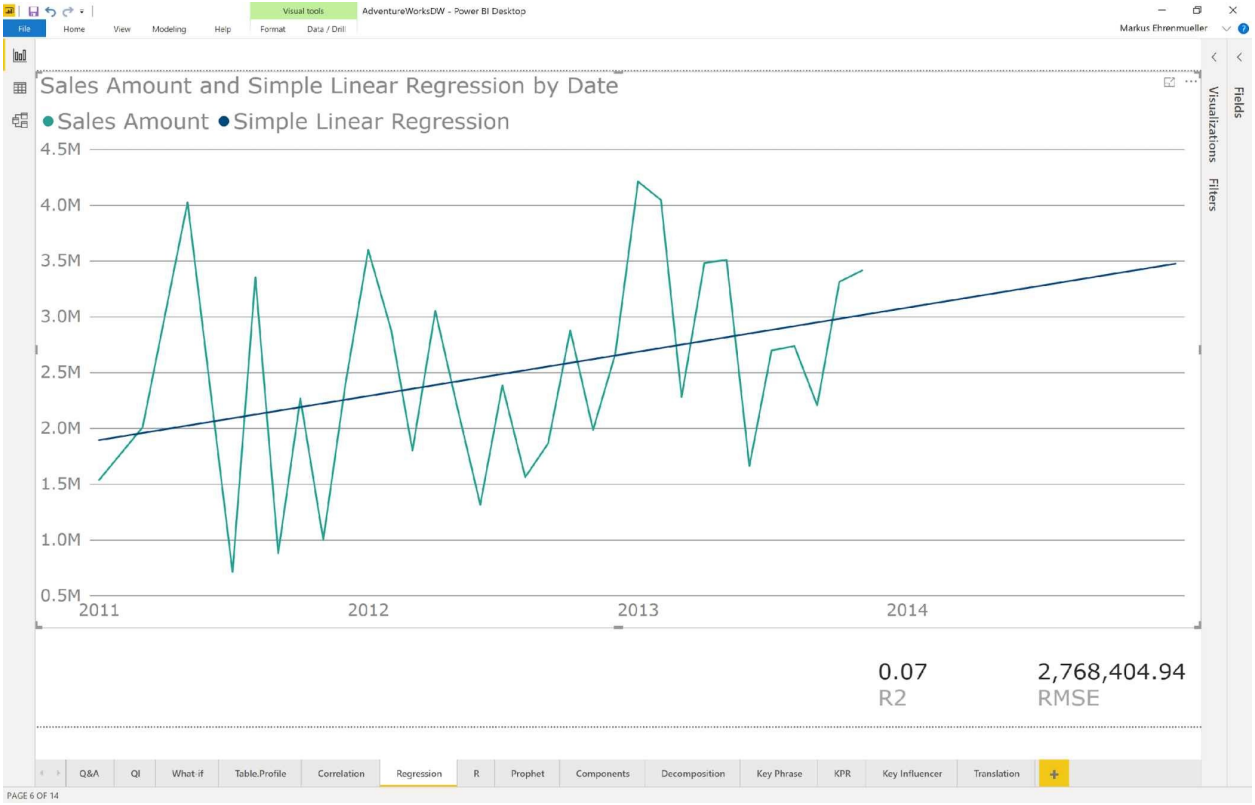


Figure 11-01: Sales Amount and Simple Linear Regression by Date

While Simple Linear Regression has its limitations and assumptions, it is simple enough to understand it with mere basic mathematical knowledge and allows you to learn about the data (in our case: the correlation between sales amount and the order date). The formula is:

```
Y = Intercept + Slope * X
```

While: `Intercept` is the amount for `Y` where the line crosses the `Y` -axis. And `Slope` states how much *Y* increases for every increase of `X` (for positive valus of `Slope`; a negative `Slope` would mean a decreasing value of `Y` for every increase of `X`).

In plain English: When `X` is zero, `Y` is calculated as the `Intercept`. This might not be an appropriate assumption in all cases (e. g. because there is not a real-

world zero-value for our order date). And `Slope` tells us, how much `Y` increases (or decreases for that matter) every single day over the order date.

The actual values for `Intercept` and `Slope` can be calculated via the following formulas:

$$\text{Slope} = \frac{N * \sum(x * y) - \sum x * \sum y}{N * \sum x^2 - (\sum x)^2}$$

$$\text{Intercept} = \frac{\sum y}{N} - \text{Slope} * \frac{\sum x}{N}$$

Now let's look how we can apply this in a useful way:

- Analytic Line
- DAX
- R Visual

Analytic Line

For certain visualisations Power BI offers us to apply Analytics. In the case of a line chart we can apply the following:

- Trend line
- Constant line
- Min line
- Max line
- Average line
- Median line
- Percentile line

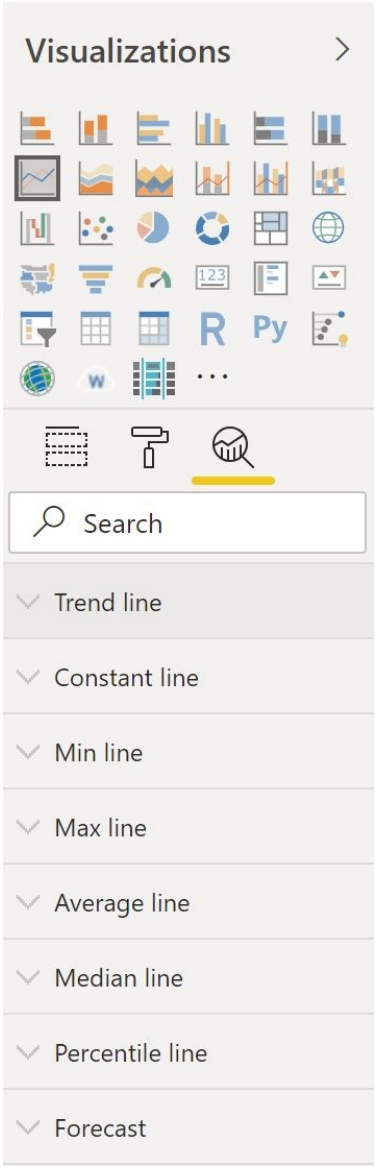


Figure 11-02: Analytic Trend Line

The first entry, trend line, automatically calculates and displays a simple linear regression line, as discussed above and as we can see in the following screenshot as a straight, dotted black line.

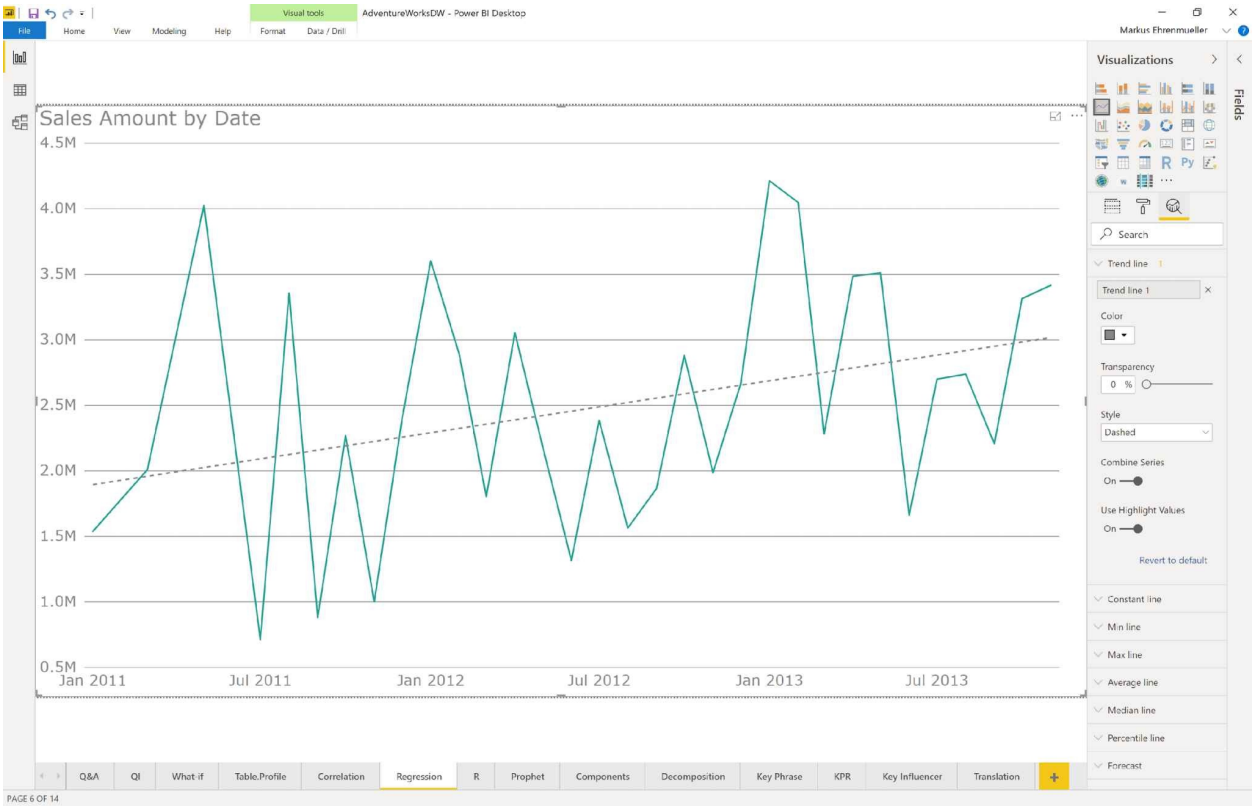


Figure 11-03: Sales Amount by Date and Analytic Trend Line

To apply the trend line, we don't even have to know what simple linear regression is or how to calculate it. That's good for that matter. The only disadvantage is: Even when we know how to calculate it, we cannot influence the calculation of the trend line in any way. If you are satisfied with the trend line how it is, then go for it. If you are not satisfied with the calculation, you can't do anything here, but make use of DAX or R, which is explained below.

Exercise: Try out the other available analytic lines as well. In what use cases they could be of help for your existing reports?

DAX

Daniil Maslyuk describes an implementation of simple linear regression in DAX in his blog (https://xxlbi.com/blog/simple-linear-regression-in-dax/). The DAX formula is quite lengthy, text-wise, as Daniil made use of variables, which helps to understand the single elements of the formula:

```
Simple Linear Regression =
/* Simple linear regression via DAX by Daniil Maslyuk
 * https://xxlbi.com/blog/simple-linear-regression-in-dax/
 */
```

```
VAR Known =
    FILTER (
        SELECTCOLUMNS (
            ALLSELECTED ( 'Date'[Date] ),
            "Known[X]", 'Date'[Date],
            "Known[Y]", [Sales Amount]
        ),
        AND (
            NOT ( ISBLANK ( Known[X] ) ),
            NOT ( ISBLANK ( Known[Y] ) )
        )
    )
VAR Count_Items =
    COUNTROWS ( Known )
VAR Sum_X =
    SUMX ( Known, Known[X] )
VAR Sum_X2 =
    SUMX ( Known, Known[X] ^ 2 )
VAR Sum_Y =
    SUMX ( Known, Known[Y] )
VAR Sum_XY =
    SUMX ( Known, Known[X] * Known[Y] )
VAR Average_X =
    AVERAGEX ( Known, Known[X] )
VAR Average_Y =
    AVERAGEX ( Known, Known[Y] )
VAR Slope =
    DIVIDE (
        Count_Items * Sum_XY - Sum_X * Sum_Y,
        Count_Items * Sum_X2 - Sum_X ^ 2
    )
VAR Intercept =
    Average_Y - Slope * Average_X
RETURN
        SUMX (
        DISTINCT ( 'Date'[Date] ),
        Intercept + Slope * 'Date'[Date]
```

```
    )
```

Figure 11-04: Simple Linear Regression in DAX

When you use the DAX measure as it is, you will end up with the very same trend line as with the analytic line in the example above, but with two exceptions. One, the trend line continues over the whole of the dates available in your date-table (even if there are not actual values available for that time-frame). In the screenshot (Figure 11-1) the blue trend line is continued until end of 2014:

Second, we can freely influence the trend line by pushing the right buttons. In the following example I introduced the following:

a) A filter for the calculation of variable *Known*. Therefore the calculation of the trend line is not based on all of the available actual sales, but limited to sales from 2013 only. As the trend for year 2013 is negative, the trend line now points down.

And b), another filter in the *RETURN* expression, as well. This filter returns values for dates after January 1 2013, only. Therefore, the trend line starts not anymore at the very left of the chart, but in year 2013. I marked the changes in bold font:

```
Simple Linear Regression 2 =
/* Simple linear regression 2 via DAX
 * based on formula by Daniil Maslyuk
 * https://xxlbi.com/blog/simple-linear-regression-in-dax/
 */
VAR Known =
    FILTER (
        SELECTCOLUMNS (
            FILTER ( ALLSELECTED ( 'Date'[Date] ); 'Date'[Date] >=
DATE(2013; 01; 01) );
            "Known[X]", 'Date'[Date],
            "Known[Y]", [Sales Amount]
        ),
        AND (
            NOT ( ISBLANK ( Known[X] ) ),
            NOT ( ISBLANK ( Known[Y] ) )
        )
    )
```

```
VAR Count_Items =
   COUNTROWS ( Known )
VAR Sum_X =
   SUMX ( Known, Known[X] )
VAR Sum_X2 =
   SUMX ( Known, Known[X] ^ 2 )
VAR Sum_Y =
   SUMX ( Known, Known[Y] )
VAR Sum_XY =
   SUMX ( Known, Known[X] * Known[Y] )
VAR Average_X =
   AVERAGEX ( Known, Known[X] )
VAR Average_Y =
   AVERAGEX ( Known, Known[Y] )
VAR Slope =
   DIVIDE (
      Count_Items * Sum_XY - Sum_X * Sum_Y,
      Count_Items * Sum_X2 - Sum_X ^ 2
   )
VAR Intercept =
   Average_Y - Slope * Average_X
RETURN
   CALCULATE (
      SUMX (
      DISTINCT ( 'Date'[Date] ),
      Intercept + Slope * 'Date'[Date]
      );
      FILTER ( 'Date'; 'Date'[Date] >= DATE(2013; 01; 01) )
   )
```

Figure 11-05: Simple Linear Regression in DAX with filters applied

Exercise: Create a DAX measure for each of the other available analytic lines (e. g. min line, max line, average line, etc.). Change the formula a) to adopt the calculation to only a certain time frame (e. g. calculating the min line for year 2013 only) and b) to adopt the length of the line to only a certain time frame (e. g. showing the min line only along 2013).

R Visual

Note: While the R visual is a standard visual in Power BI and automatically available, you need a local instance of R installed on your computer. Please follow instructions at https://mran.microsoft.com/ on how to install (a free copy) of R. As I make use of libraries ggplot2 , scales and (later) tm in my examples, make sure to install those libraries as well. I would recommend to install an R IDE (e. g. R studio) and to run the following statement: " install.packages("ggplot2", "scales", "tm") ".

In the following example, I created a *R visual*, which runs the script listed below. I make use of the ggplot2 library to visualize the actual sales amount over date (green line). Via function stat_smooth I let ggplot() calculate a simple regression line (" method=lm "), shown as blue line. By default, ggplot is showing a confidence interval (grey area along the blue line) as well, which can be turned off (by adding parameter " se=FALSE " to stat_smooth).

Figure 11-06: Simple Linear Regression in an R Visual

```
# Simple Linear Regression in R Visual

# data cleaning
FactResellerSales <- dataset
```

```
colnames(FactResellerSales) <- c("OrderDate", "SalesAmount")
FactResellerSales <- FactResellerSales[FactResellerSales$OrderDate!="",]
FactResellerSales$OrderDate <- as.POSIXct(FactResellerSales$OrderDate)

# adding a regression line
library(ggplot2)
library(scales) #generic plot scaling methods
ggplot(FactResellerSales,
       aes(x = OrderDate,
           y = SalesAmount)) +
  geom_line(color = 3) +
  theme(text = element_text(size = 18)) +
  scale_y_continuous(limits = c(0, 5000000), labels = comma) +
  theme(legend.position = "none") +
  labs(x = "Order Date",
       y = "Sales Amount") +
  stat_smooth(method=lm) # linear model / linear regression
```

Figure 11-07: R script do display sales amount over date and a simple linear regression line

Exercise: With the search engine of your choice find out which other methods stat_smooth allows. Try them out and evaluate if those methods give a better fitting line.

Summary

In this chapter we looked into different implementations of simple linear regression. We added a trend line to our line chart via Power BI's analytic features. This is easy to apply, but does not offer much flexibility, according to the calculation and display of the line. To achieve flexibility, we then calculated simple regression via the help of both, DAX and R. The implementation in DAX is straight forward, if you are used to writing DAX. The implementation in R was done with stat_smooth function available as part of the ggplot library.

Text Mining

In this example we are going to analyse the product description column. The description consists of shorter and longer texts. I combined the product's *EnglishProductName* and *EnglishProductDescription* into a new column *EnglishProductNameAndDescription*, as the description column might be empty for some of the columns, while there is always a product name.

If you add the description to an ordinary table visual, it is hard to compare different products or get an idea how those descriptions differ between different product categories. Therefore, we are going to "mine" the content of the column.

Word Cloud

I assume, that the more important a term in a text is, the more often it will appear in the text. A word cloud visual counts how often a term is mentioned. Terms with a higher count are printed in a bigger font size and appears more towards the centre than the others. I've included a screenshot below:

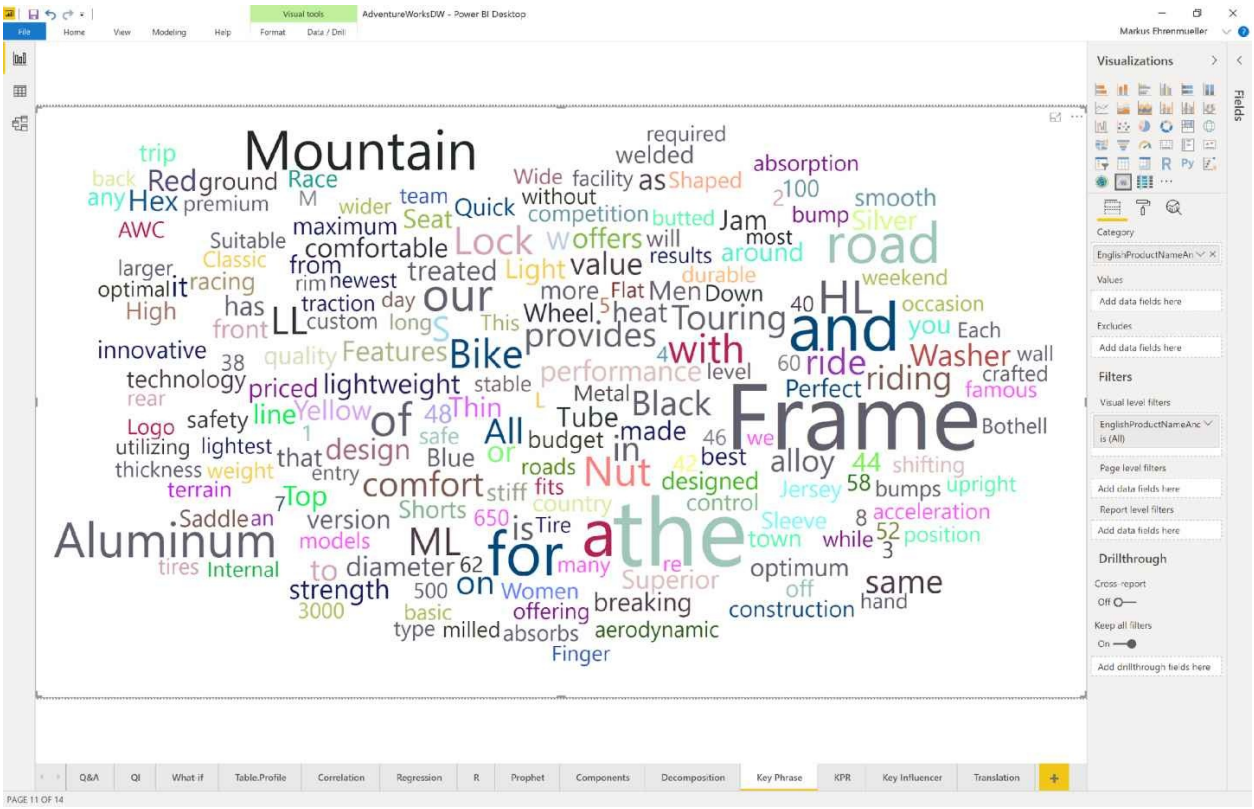


Figure 11-08: Word cloud of column EnglishProductNameAndDescription

This is easy to apply but comes with strings attached: In the English language certain words, like "and", "the", "a", "for", "on", "of", or "with" are very common. Therefore, they are also more common in the product descriptions. But

as they are not special for a certain product category, we would rather like to remove them from the word cloud. That's what we are going to do with the help of the following technologies:

- Azure Cognitive Services
- Azure Machine Learning
- R Visual

Azure Cognitive Services

Azure Cognitive Services is Microsoft's offer for so-called pre-trained models. Microsoft's data science team is developing and publishing an ever-growing amount of different services, which are used internally as well (e. g. to localize the online documentation at docs.microsoft.com). "Pre-trained" means, that Microsoft took care of tuning the models. This guarantees a high level of quality for lot of use cases. Currently Azure Cognitive Services offers different features like decision-making, vision, speech, search, and language. The latter offers services like Text Analytics, Translator Text, QnA Maker, Language Understanding, and Immersive Reader. For our use case we will leverage "key phrase", which is part of the "Text Analytics" offer.

To make use of Azure Cognitive Services you need to go through the following steps:

- Sign-in to portal.azure.com and add "Cognitive Services" to one of your subscriptions.
- For later reference we will need one of the both generated keys and the API endpoint. Both can be obtained by opening the newly created Cognitive Services.
- I created a parameter in Power Query for both: *apikey* and *endpoint*. This makes it easier later in case you have several references to both and want to change them.

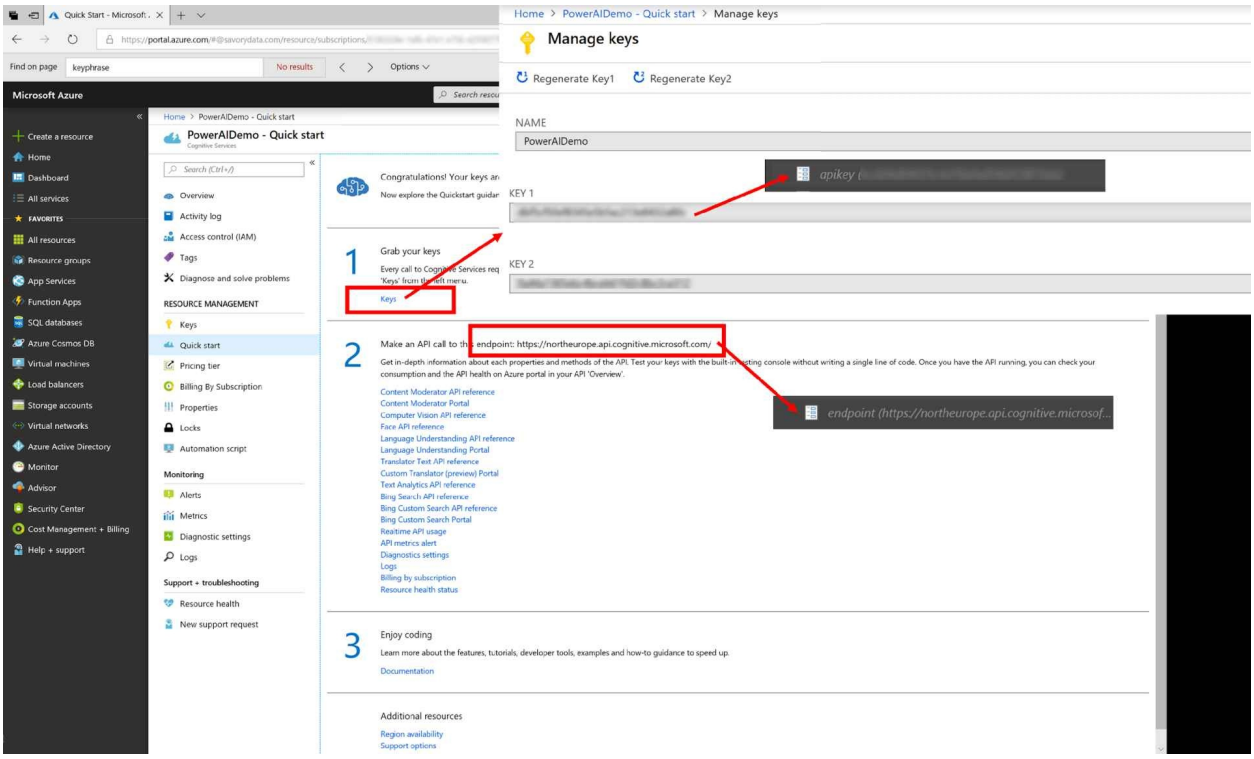


Figure 11-09: Where to find API key and endpoint of Cognitive Services in Azure's portal

Create a function in Power Query by creating a new query and choosing "Blank query". Then you insert the following script in the *Advanced editor*. I took this script from Microsoft's online documentation and changed the assignment of apikey and endpoint to use the Power Query parameters, created in the step above. The script is accepting one parameter (text), converting it into a JSON format, sending it to Cognitive Services by using apikey and endpoint and converting the returned JSON value back into a textfield. Make sure to rename the query to "Keyphrase API".

```
// https://docs.microsoft.com/en-us/azure/cognitive-services/text-analytics/tutorials/tutorial-power-bi-key-phrases
// Returns key phrases from the text in a comma-separated list

(text) => let
   apikey      = apikey,
   endpoint    = endpoint & "/keyPhrases",
   jsontext    = Text.FromBinary(Json.FromValue(Text.Start(Text.Trim(text), 5000))),
   jsonbody    = "{ documents: [ { language: ""en"", id: ""0"", text: " & jsontext & " } ] }",
```

```
    bytesbody   = Text.ToBinary(jsonbody),
    headers     = [#"Ocp-Apim-Subscription-Key" = apikey],
    bytesresp   = Web.Contents(endpoint, [Headers=headers,
Content=bytesbody]),
    jsonresp    = Json.Document(bytesresp),
    keyphrases  = Text.Lower(Text.Combine(jsonresp[documents]{0}
[keyPhrases], ", "))
in  keyphrases
```

Figure 11-10: Power Query function which accepts a text parameter and returns the key phrases

This Power Query function we can now apply on any column in any Power Query. Select a column of type text and then select “Add column” in the menu and “Invoke Custom Function” in the ribbon. Give the resulting column a useful name (e. g. “keyphrases”) and select “Keyphrase API” as “Function query”. As soon as you click OK every single row of your Power Query will be sent over to Cognitive Services and the parsed key phrases are returned back to Power Query.

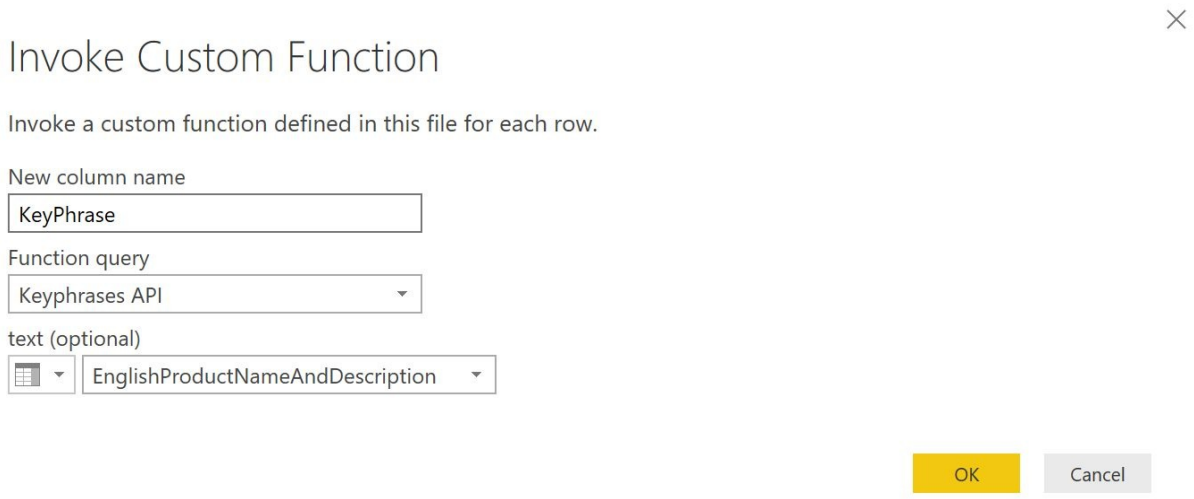


Figure 11-11: Invoke custom function “Keyphrases API” on column EnglishProductNameAndDescription

Please be aware of two facts: One, costs will be charged to your Azure subscription every time you refresh Power Query. Please refer to “pricing tier” to find out about how much. In my subscription I will currently be charged EUR 0.84 per 1000 calls to the API, but there are trial subscriptions available which

give you a limited amount of credits free of charge.
Second, the content of the column you are invoking Cognitive Services on is sent over the internet to Azure. In case this would violate any privacy restrictions, please consider running Cognitive Services in a container on your premises instead.

In cases where you want to build upon a pre-trained model or come up with a model on your own, you must consider one of the following options: Azure Machine Learning or R.

Azure Machine Learning

Azure Machine Learning allows you to build your own experiments by using different building blocks and deploying the experiment as a web service which can be called by your application. We will build a Machine Learning experiment, deploy it and then call it in Power Query now.

In the first example we will make use of a building block “Extract Key Phrases from Text”. This is based on the same pre-trained model as we used above in Cognitive Services. The extracted key phrases will therefore be the same. Azure Machine Learning gives though the option to clean and transform the data before and after the building block, which I will not do in this first example to keep it simple. (In the next chapter we will build our own algorithm to extract key phrases.)

As you can see in the screenshot below, I used “Book reviews from Amazon” as a basis for my experiment. This dataset, and many more, are available by default so you can easily start experimenting. Then I used the step “Edit metadata” to rename the columns of the dataset to “ProductKey” and “EnglishProductNameAndDescription” to match the column names of my dataset I will later use in Power Query. “Partition and Sample” allows me to combine both, “Book reviews from Amazon” and “Web service input” into one set of rows. Then I “Extract Key Phrases from Text” and combine the discovered keyphrases with the original two columns “ProductKey” and “EnglishProdctNameAndDescription”. Finally, “Web services output” makes sure to send all three columns back to the caller of the web service.

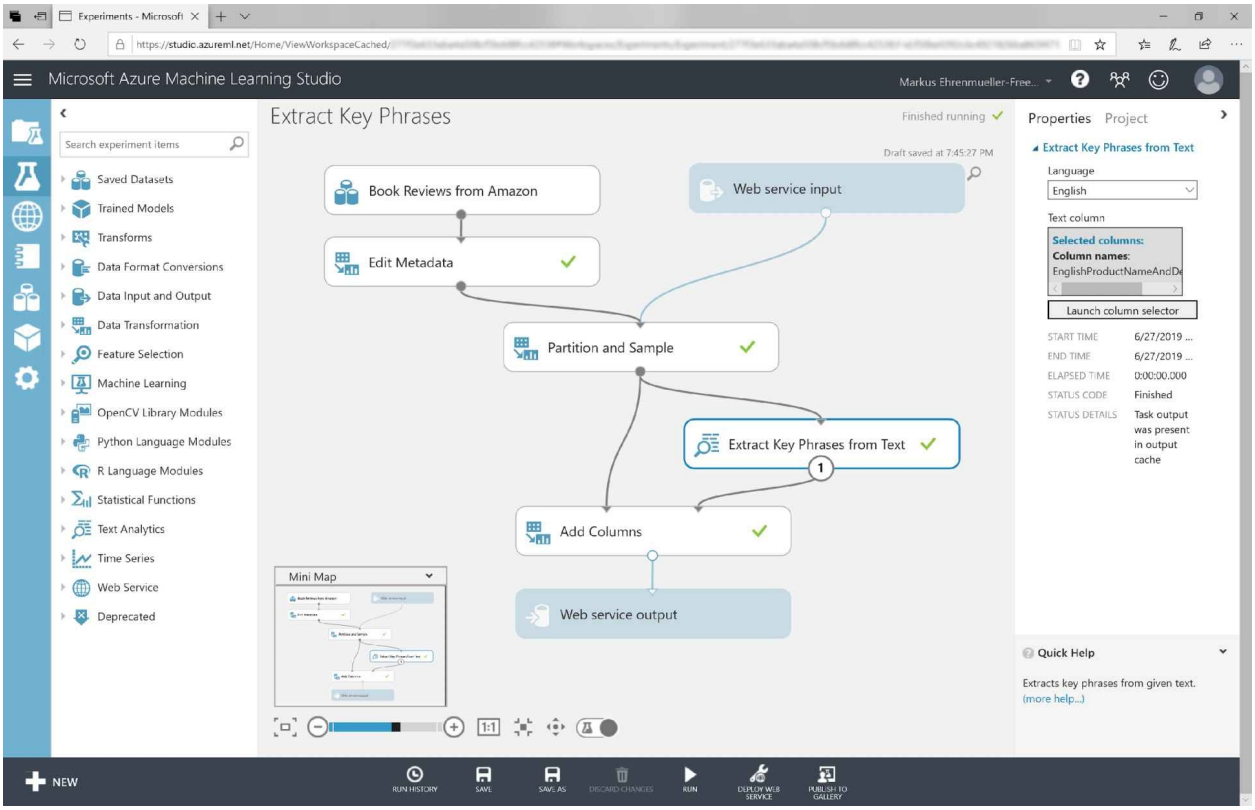


Figure 11-12: An Experiment in Azure Machine Learning Studio to extract key phrases

Before you can finally press the button “deploy web service”, you first have to press button “run” to run the experiment once.

When in “experiment view” () the blue lines and the web service component are ignored, so the experiment is not waiting for a web service call or returning data. When in “web service view” (), on the other hand, the “Book reviews from Amazon” and the “Edit Metadata” is ignored, so the web service result is not influenced from the data of the experiment.

Before we can incorporate a call to our new web service in Power Query with the method I will describe in a minute, we must gather two important parameters of the web service: API key of the web service and the request URI of the API. I created Power Query parameters for both: *WebServiceAPIKey* and *RequestURI*.

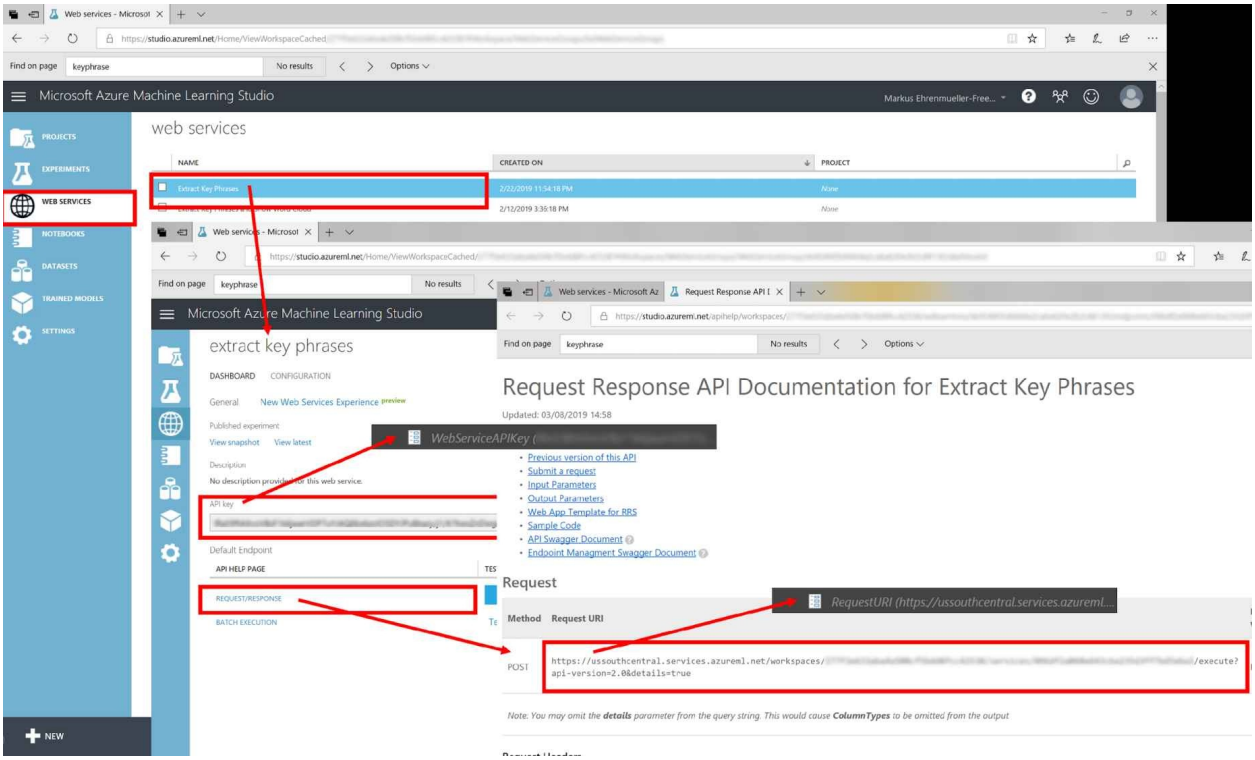


Figure 11-13: Where to find API key and request URI of Azure Machine Learning web service

While different possibilities of calling a Azure ML web service from Power Query are available, I prefer Gerhard Brückl's approach (https://blog.gbrueckl.at/2016/06/score-powerbi-datasets-dynamically-azure-ml/). Basically, he wrote three Power Query functions: One to convert Power Query data into JSON (*ToAzureMLJson*), one to convert a JSON into a Power Query table (*AzureMLJsonToTable*) and one to call Azure ML's API and making use of the first two functions (*CallAzureMLService*). You can read more details about that in his excellent blog post.

The rest of the steps are like those described in the previous chapter of applying a function on a column in Power Query. The function *CallAzureMLService* takes the web service's API key and the request URI as a parameter (just select the Power Query Parameter for that). Furthermore, we must pass in the name of a table, which must consist of exactly those two columns we defined for the input of the web service, with exactly the same name and data type. The function then returns the two columns plus an extra one, containing the key phrases.

Please be aware of two facts: Costs will be charged to your Azure subscription every time you refresh Power Query. Please refer to "pricing tier" to find out

about how much. I used a free subscription of Azure ML, which allows for only a limited amount of calls, which was sufficient for my sample.
Second the content of the whole table selected in the third parameter of the Power Query function *CallAzureMLService* is sent over the internet to Azure. Currently there is no on-premise alternative (e. g. containers) available for Azure ML.

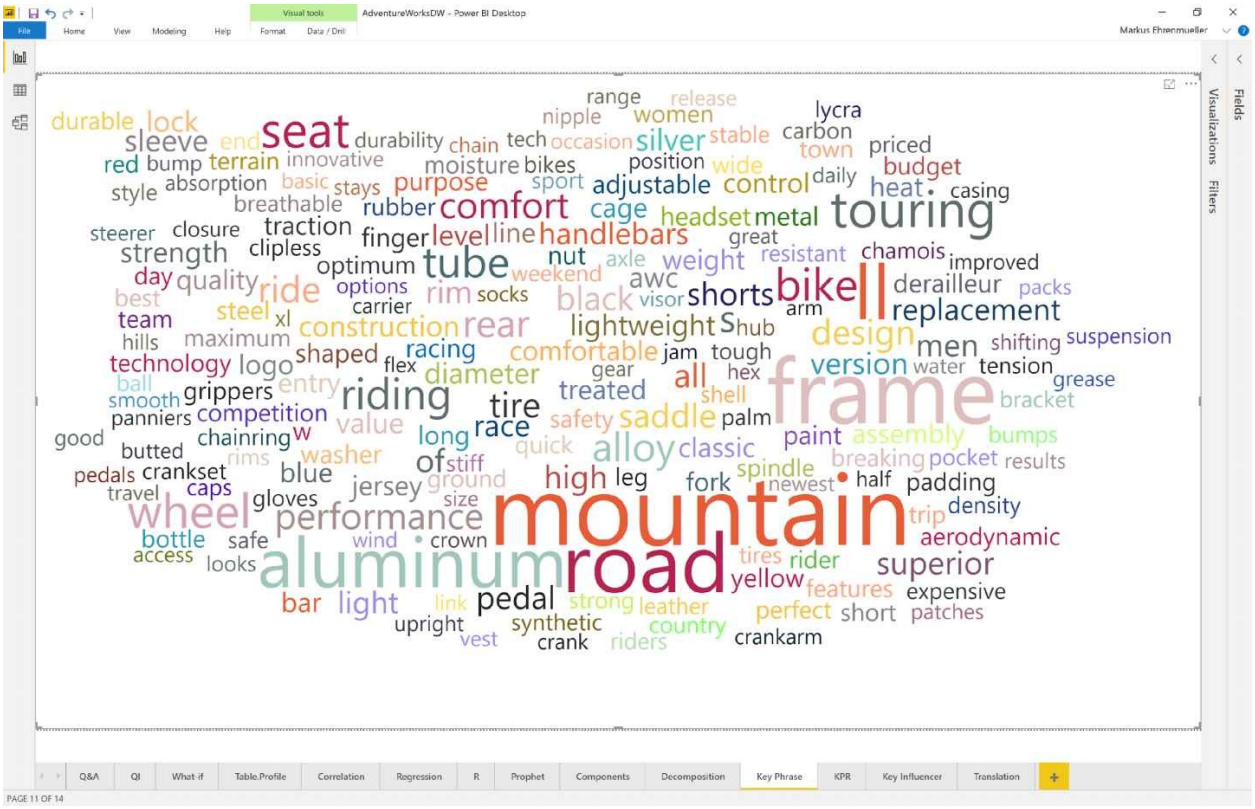


Figure 11-14: Word cloud based on Cognitive Services' key phrases

R in Azure Machine Learning

In cases the available machine learning algorithms in Azure Machine Learning are not sufficient for your use case you can always plug in R code into it. You can use R to create both, a model, and to run a R script in Azure Machine Learning. The following example will show you, how you can extract key phrases via R. I copied the experiment from above and exchanged the "Extract Key Phrases from Text" with "Execute R Script". You can find the R script here:

```
# Map 1-based optional input ports to variables
dataset <- maml.mapInputPort(1) # class: data.frame

library(tm)
```

```
corpus_clean <-
VCorpus(VectorSource(dataset$EnglishProductNameAndDescription))
corpus_clean <- tm_map(corpus_clean, content_transformer(tolower)) # apply r-standard function tolower() to content
corpus_clean <- tm_map(corpus_clean, stemDocument)
corpus_clean <- tm_map(corpus_clean, stripWhitespace) # eliminate unneeded whitespace
corpus_clean <- tm_map(corpus_clean, removeNumbers) # apply tm's removeNumbers to content
corpus_clean <- tm_map(corpus_clean, removeWords, stopwords()) # remove stop words
corpus_clean <- tm_map(corpus_clean, removePunctuation) # remove punctuation
corpus_clean <- tm_map(corpus_clean, stemDocument)
corpus_clean <- tm_map(corpus_clean, stripWhitespace) # eliminate unneeded whitespace

# Select data.frame to be sent to the output Dataset port
df <- data.frame(text = sapply(corpus_clean, paste, collapse = " "),
stringsAsFactors = FALSE)
maml.mapOutputPort("df")
```

Figure 11-15: R script to fetch Azure ML's input, find key phrases and return them as output

After running the experiment and deploying it as a web service you have to then grab the API key of the web service and the request URI of the API and apply Power Query function *CallAzureMLService* exactly as described in the previous chapter.

Before you write your own R scripts to run in Azure Machine Learning, please check if the libraries you want to use are available in Azure. Find a list of available libraries here: https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/r-packages-supported-by-azure-machine-learning.

Again: This is an easy way to share a functionality across different Power BI files and can be applied in Excel or your own application as well. Costs apply for every call of the web service and data is exposed over the internet. If you want to avoid those costs and the exposure over the internet you can incorporate the R

script directly as a Power Query transformation step, as described in the following chapter.

R as a Power Query Transformation

We've already seen in the chapter about linear regression that we can write R code to run in an R visual. Power BI offers two more possibilities, to directly inject R code: R source and R transformation in Power Query. Let's check out R transformation in the next example.

In the Power Query window find the table containing the column "EnglishProductNameAndDescription" and select "Transform" in the menu and select "Run R script" in the Ribbon. Then copy & paste the script from above, where I only removed the first and last line of code, which controlled the communication between Azure ML and R. In R transformation there is no need for calling special functions (like maml.mapInputPort or maml.mapOutputPort), as all the columns and rows available in the current Power Query are injected as a data frame with name "dataset" into the script:

```
library(tm)
corpus <-
VCorpus(VectorSource(dataset$EnglishProductNameAndDescription))
corpus_clean <- tm_map(corpus, content_transformer(tolower)) # apply r-standard function tolower() to content
corpus_clean <- tm_map(corpus_clean, stemDocument)
corpus_clean <- tm_map(corpus_clean, stripWhitespace) # eliminate unneeded whitespace
corpus_clean <- tm_map(corpus_clean, removeNumbers) # apply tm's removeNumbers to content
corpus_clean <- tm_map(corpus_clean, removeWords, stopwords()) # remove stop words
corpus_clean <- tm_map(corpus_clean, removePunctuation) # remove punctuation
corpus_clean <- tm_map(corpus_clean, stemDocument)
corpus_clean <- tm_map(corpus_clean, stripWhitespace) # eliminate unneeded whitespace
```

Figure 11-16: R script to find key phrases

This solution runs completely locally – there are no additional costs during refresh and no data leaves your laptop. But this comes on the price of less

reusability, as you have to duplicate the code in every Power BI file. Applying changes to the code can be challenging, as you must maintain different copies of the code.

The R code is executed only once during refresh and cannot be influenced by filters. If the R script needs to be parametrized by user selection, we can us a R visual instead.

R Visual

You've already seen a R visual above, when we used ggplot to calculate and draw the regression line. We can do the same for our key phrases and the word cloud.

The code below to extract the key phrases is the same as above, except that it has lines of code added to create a word cloud visual:

```
library(tm)
corpus <-
VCorpus(VectorSource(dataset$EnglishProductNameAndDescription))
corpus_clean <- tm_map(corpus, content_transformer(tolower)) # apply r-standard function tolower() to content
corpus_clean <- tm_map(corpus_clean, stemDocument)
corpus_clean <- tm_map(corpus_clean, stripWhitespace) # eliminate unneeded whitespace
corpus_clean <- tm_map(corpus_clean, removeNumbers) # apply tm's removeNumbers to content
corpus_clean <- tm_map(corpus_clean, removeWords, stopwords()) # remove stop words
corpus_clean <- tm_map(corpus_clean, removePunctuation) # remove punctuation
corpus_clean <- tm_map(corpus_clean, stemDocument)
corpus_clean <- tm_map(corpus_clean, stripWhitespace) # eliminate unneeded whitespace

library(wordcloud)
library(RColorBrewer)
wordcloud(corpus_clean, min.freq = 1, random.order = FALSE,
colors=brewer.pal(9, "BuGn"))
```

Figure 11-17: R script to find key phrases and plot them in a word cloud

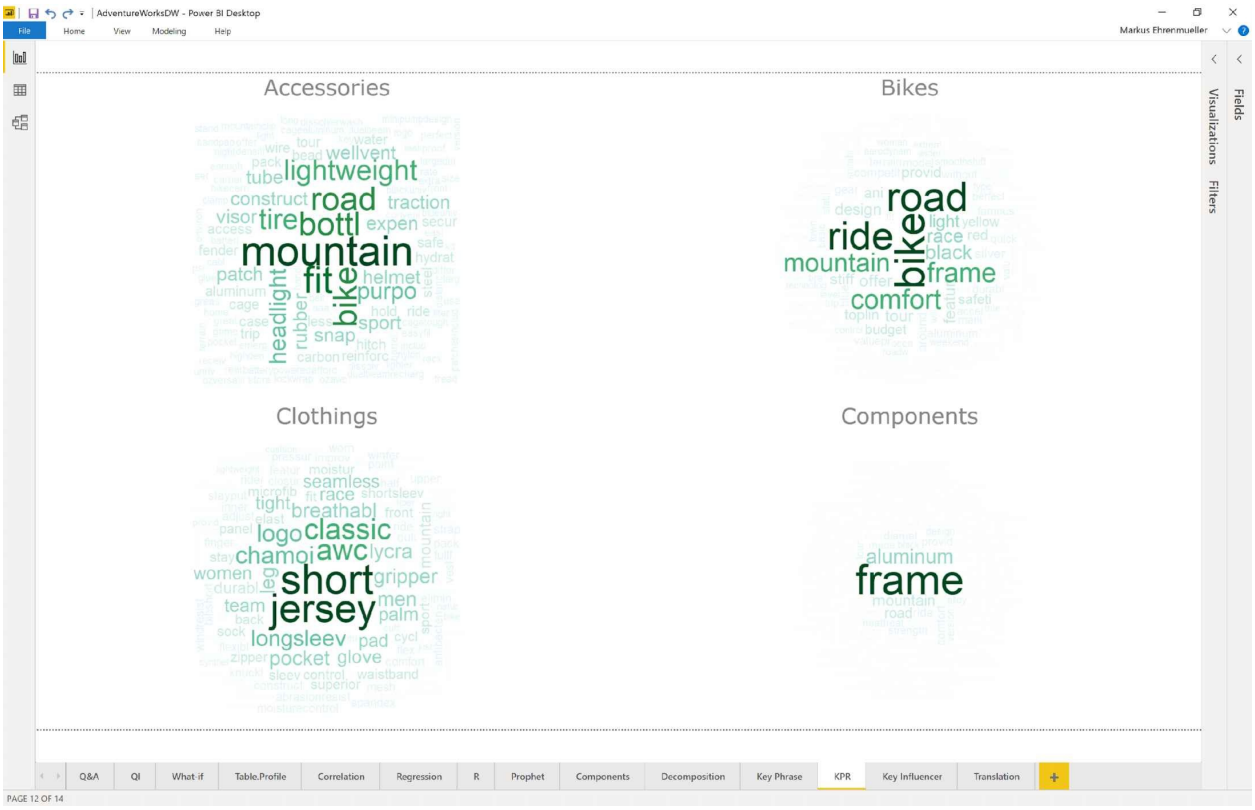


Figure 11-18: Word cloud based on key phrases discovered by R's tm package

Summary

In this chapter we have seen different possibilities to extract key phrases out of a free text column (the concatenated product name and description). Every solution had its own strengths and weaknesses, as shown in the following table:

	Cognitive Services	**Azure ML Key Phrase**	**Azure ML R**	**Power Query R Transformation**	**Power BI R visual**
Additional Costs	Apply	Apply	Apply	No	No
Data is sent over the internet	Yes *)	Yes	Yes	No	No
Reuseability	Yes	Yes	Yes	No	No
Calculation	Data refresh	Data refresh	Data refresh	Data refresh	Visual refresh
User filter	No	No	No	No	Apply
Pre-trained vs. self-trained Model	Pre-trained	Pre-trained	Self-trained	Self-trained	Self-trained

*) You can run Cognitive Services in a container on premises to avoid exposure of data.

About the Author

Markus Ehrenmüller-Jensen is the founder of Savory Data and works as a project leader, trainer & consultant for data engineering, business intelligence, and data science since 1994. He is an educated software-engineer, graduated business educator and professor for databases and project engineering at HTL Leonding (technical college) and certified as MCSE Data Platform, MCSE Business Intelligence, and MCT. Markus speaks regularly on international conferences (eg. PASS Summit, SQLBits, SQL Saturdays, SQL Days, Power BI World Tour, ...) and writes articles for well-known journals. In 2013 he co-founded SQL PASS Austria and in 2015 Austria PUG and organizes SQL Saturdays in Austria since 2014. For his technical leadership in the community he was awarded as a Microsoft Data Platform MVP since 2017.

Chapter 12: Automated Machine Learning in Power BI

Author: Ashraf Ghonaim

Microsoft is committed to democratizing AI through its products by making automated machine learning accessible through Power BI, so that Data Analysts and BI professionals can also take advantage of machine learning and become Citizen Data Scientists.

With Automated Machine Learning (AutoML), the data science behind the creation of ML models is automated by Power BI, with precautions to ensure model quality, and visibility to have full insight into the steps used to create the ML model.

What is Machine Learning (ML)?

Among so many definitions, Machine Learning can be defined as: "The field of study that gives computers the ability to learn without being explicitly programmed” - Arthur Samuel (1959)

ML algorithms learn from the historical data to build models that predict the future. ML at its heart is a probabilistic not a deterministic science because of generalization based on probabilities. That means any ML model has a margin of error and accuracy level.

The role of a data scientist is to come up with the model that has the highest level of possible accuracy of predicting the outcome of the new unseen data.

To be able to achieve that, the data scientist has to split the data to training dataset and testing dataset. Then, try several ML algorithms and tweak so many parameters based on their human expertise to train multiple models. The data scientist compares the results of different trained models using the testing dataset that includes unseen labeled data and finally picks the model that achieved the highest level of accuracy.

What are the challenges of Traditional ML?

Traditional ML model development process is resource-intensive, requiring significant domain knowledge and time to produce and compare dozens of models.

Data scientists and developers face a series of sequential and interconnected decisions along the way to achieving "magic" machine learning solutions. For example, should they transform the input data, and if so, how – by removing nulls, rescaling, or something else entirely? What machine learning algorithm would be best - a support vector machine (SVM), logistic regression, or a tree-based classifier? What parameter values should they use for the chosen classifier – including decisions such as what the max depth and min split count should be for a tree-based classifier? And many more.

Ultimately, all these decisions will determine the accuracy of the machine learning pipeline - the combination of data pre-processing steps, learning algorithms, and hyperparameter settings that go into each machine learning solution.

What is Automated Machine Learning (AutoML)?

Automated machine learning, also referred to as AutoML, is the process of automating the time consuming, iterative tasks of ML model development process. It allows data scientists, analysts, and developers to build ML models with high scale, efficiency, and productivity all while sustaining model quality.

AutoML democratizes the ML model development process, and empowers its users, no matter their data science expertise, to identify an end-to-end machine learning pipeline for any problem.

Data scientists, analysts and developers across industries can use AutoML to:

- Implement machine learning solutions without extensive programming knowledge
- Save time and resources
- Leverage data science best practices
- Provide agile problem-solving

AutoML empowers Power BI users, with or without data science expertise, to identify an end-to-end machine learning pipeline for any problem, achieving higher accuracy while spending far less of their time. It enables a significantly larger number of experiments to be run, resulting in faster iteration towards production-ready intelligent experiences.

AutoML is based on a breakthrough from Microsoft Research. The approach combines ideas from collaborative filtering and Bayesian optimization to search an enormous space of possible machine learning pipelines intelligently and efficiently. It's essentially a recommender system for machine learning pipelines. Similar to how streaming services recommend movies for users, AutoML recommends machine learning pipelines for data sets.

Automated Machine Learning (AutoML) in Power BI

There are several options to do Machine Learning in Power BI:

1- Python/R scripts
2- Azure Machine Leaning Studio (Web Service API)
3- Azure Machine Learning Services Integration
4- Built-in AutoML (which is the scope of this chapter)

AutoML is done in Power BI through dataflows in Power BI Service in the cloud. Dataflows offers self-serve data prep for big data. AutoML enables you to leverage your data prep effort for building ML models, right within Power BI. Dataflows is a simple and powerful ETL tool that enables analysts to prepare data for further analytics. You invest significant effort in data cleansing and preparation, creating datasets that can be used across your organization. AutoML enables you to leverage your data prep effort for building ML models directly in Power BI.

AutoML in Power BI dataflow allows to build ML models with clicks, not code, using just their Power BI skills. With AutoML, the data science behind the creation of ML models is automated by Power BI, with precautions to ensure model quality, and visibility to have full insight into the steps used to create your ML model.

AutoML for dataflows enables data analysts to train, validate and invoke ML models directly in Power BI. It includes a simple experience for creating a new ML model where analysts can use their dataflows to specify the input data for training the model. The service automatically extracts the most relevant features, selects an appropriate algorithm and tunes and validates the ML model. After a model is trained, Power BI automatically generates a report that includes the results of validation that explains the performance and results to analysts. The model can then be invoked on any new or updated data within the dataflow.

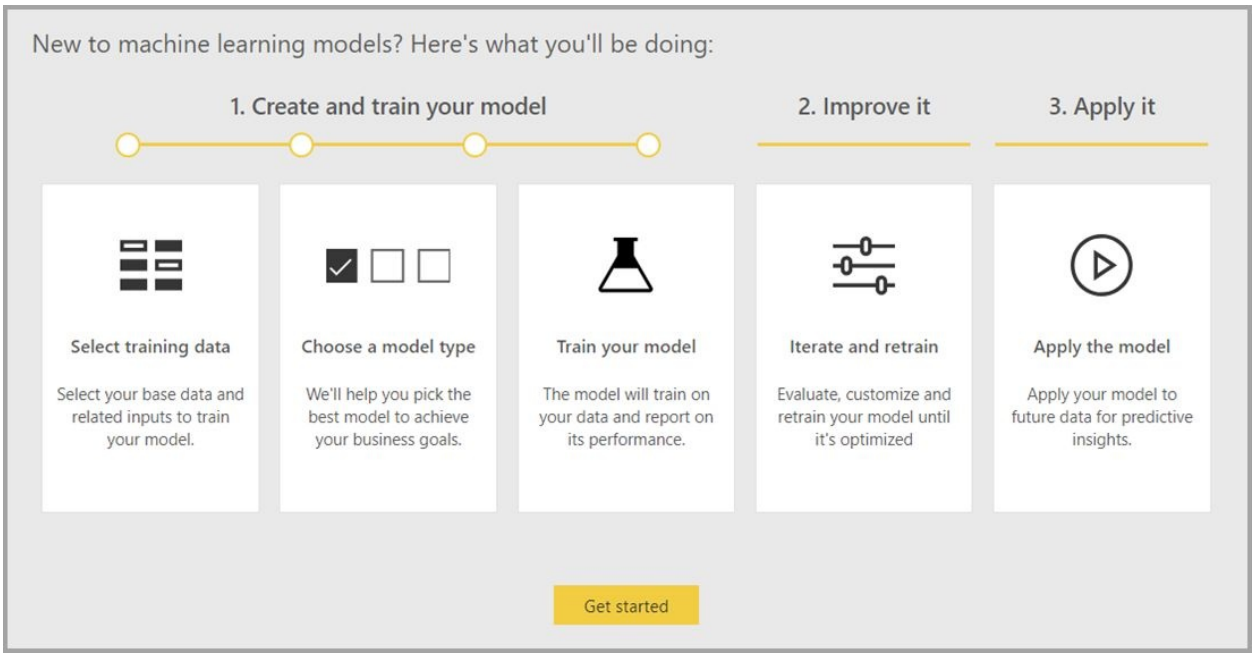


Figure 12-01: AutoML Getting Started

AutoML in Power BI integrates AutoML from the Azure Machine Learning service to create your ML models. However, you don't need an Azure subscription to use AutoML in Power BI. The process of training and hosting the ML models is managed entirely by the Power BI service.

AutoML is available for dataflows in workspaces hosted on Power BI Premium and Embedded capacities only. To be able to use it, you have to turn it on first in the premium capacity.

Enabling AutoML in your Power BI Premium Subscription

One of the goals of premium capacity is to give customers a fine-grained control of exactly what is running in each capacity. Each capacity has dedicated resources assigned, is isolated from others, and can only run generally available workloads by default. So, if there is a mission critical app running in a capacity, administrators are able to constrain other functionality to get the performance they need. For workloads in preview to run in a capacity, the capacity administrator needs to enable them and configure the maximum memory percentage available to it.

Power BI AI is a *capacity workload* and must be enabled before the features can be used. Note that this capacity workload includes Cognitive Services and AutoML. All other AI features such as Key Influencers and Quick Insights do not require a premium subscription.

You must be the Power BI Administrator or Capacity Administrator to be able to enable the Power BI AI capacity workload. Select the gear in the top right panel and click on “Admin Portal”:

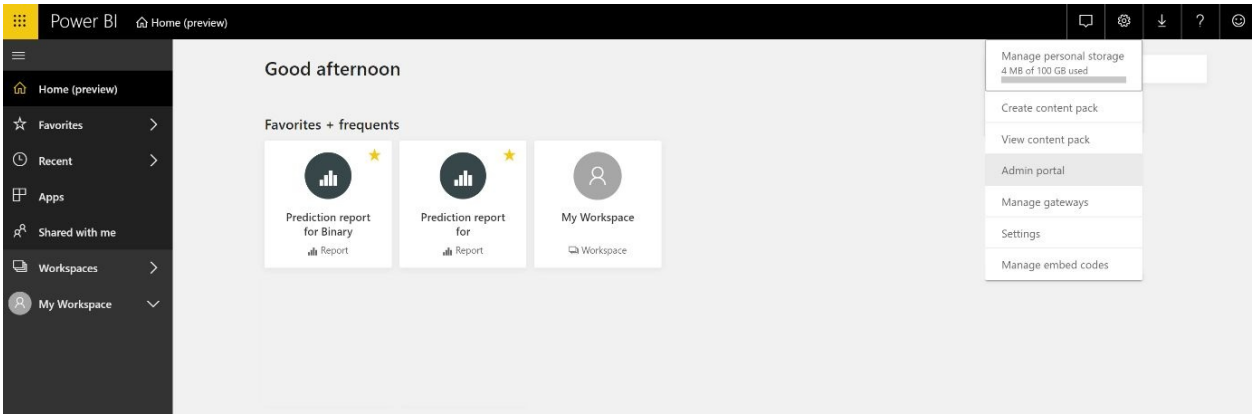


Figure 12-02: Admin Portal

Then click on the Capacity Settings on the left-hand panel on the subsequent page:

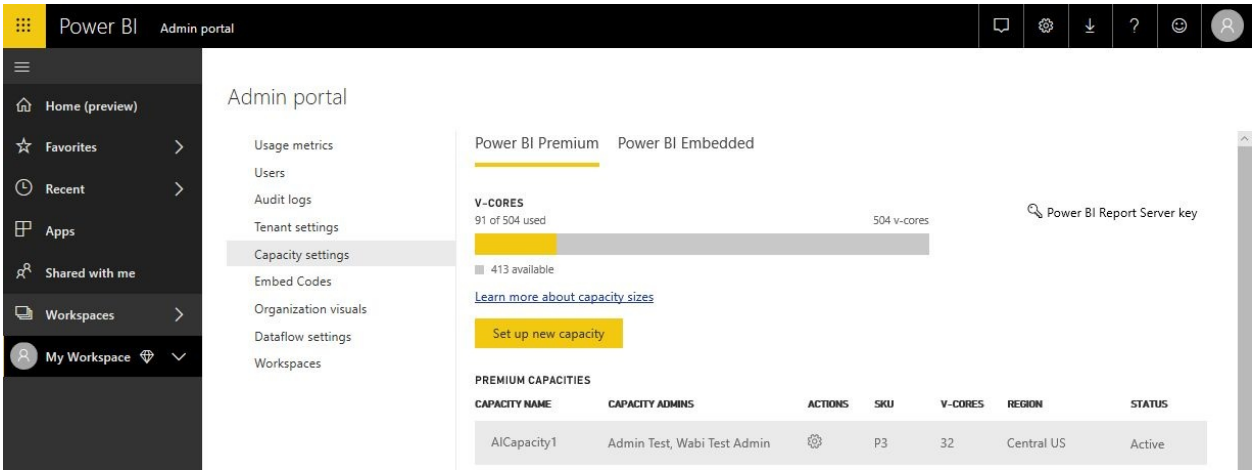


Figure 12-03: Capacity Settings

Click on the capacity where the workload is to be enabled. In the page above, it is “AICapacity1”:

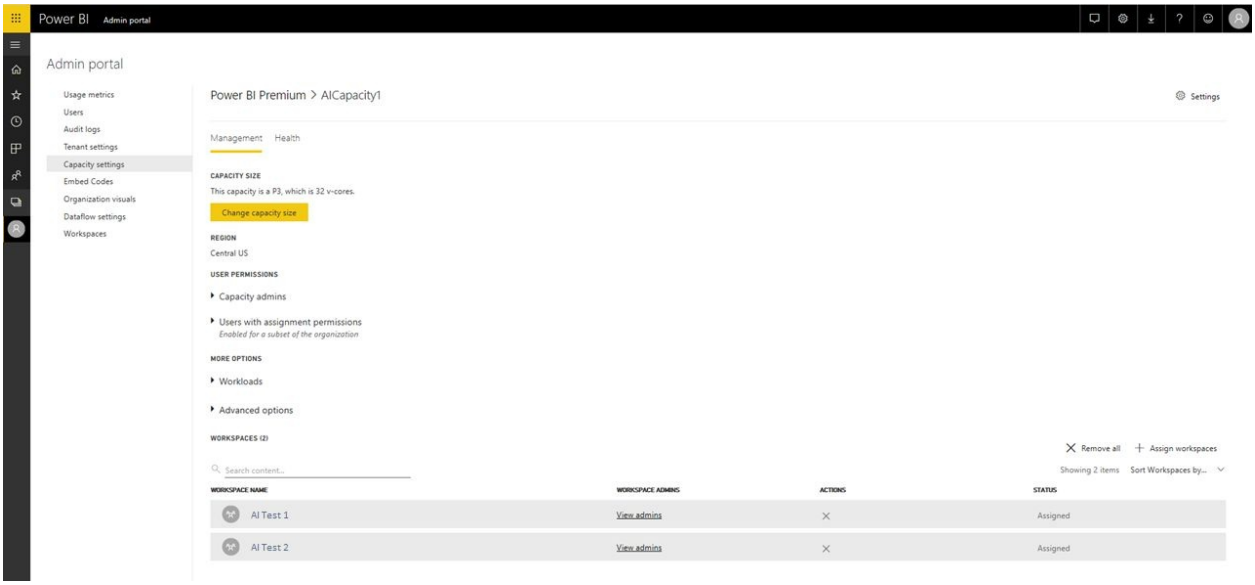


Figure 12-04: Capacity Workloads

Note that you can assign workspaces to this capacity after it is configured the way you want.

Then click “Workloads” under “More Options” and turn both Dataflows and AI (Preview) to On and set the maximum memory:

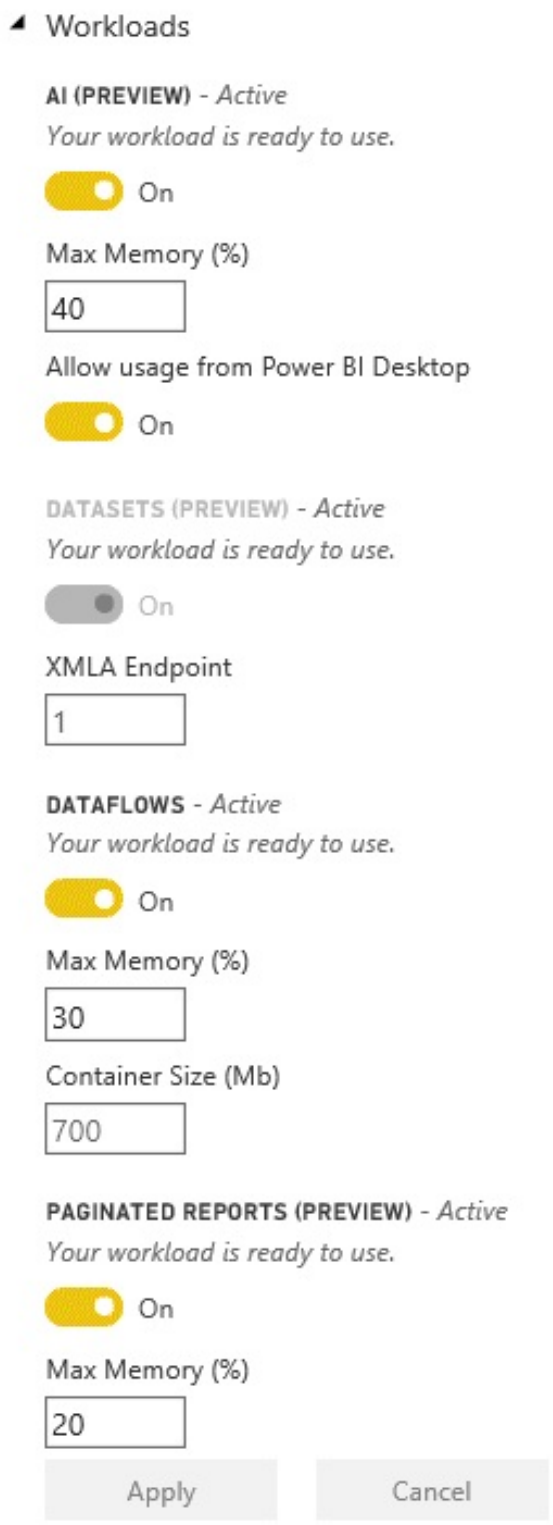


Figure 12-05: Dataflows and AI (Preview)

AutoML performs numerous AI calculations in memory. While the memory needed depends on data size and content, it is a good idea to give a liberal

amount of memory for the AI workload. Our own testing was on 5GB max memory or higher (or 100% on an A2 node).

The impact of enabling each workload is that capacity is now shared – memory used for one workload may not be available for others.

After this is applied, Cognitive Services and AutoML will be available in all of the workspaces in the selected premium capacity.

Creating an AutoML Model in Power BI

In the current preview, AutoML enables you to train ML models for Binary Prediction, Classification, Regression and Forecasting (coming soon).

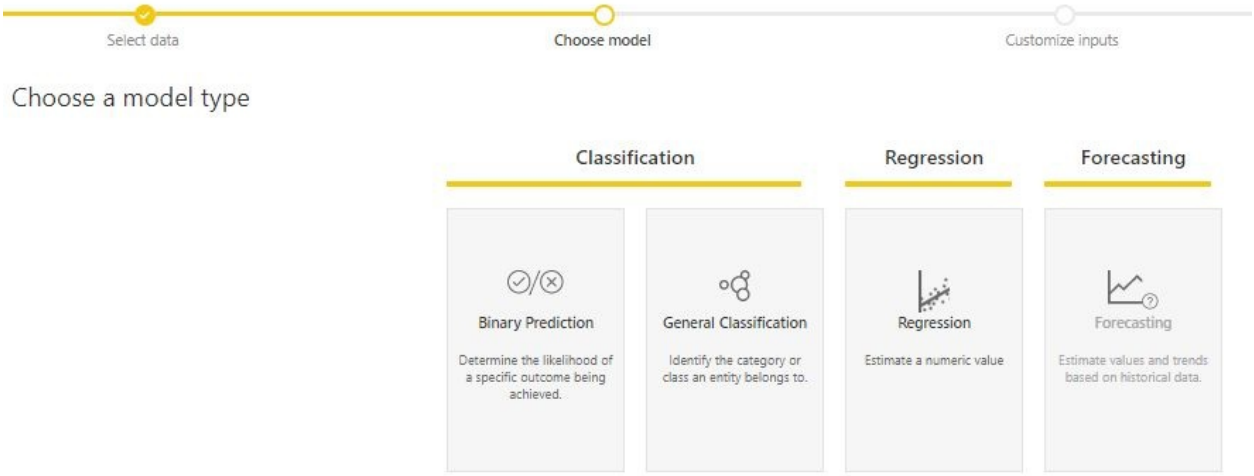


Figure 12-06: Model Types

Binary Prediction, Classification, and Regression models for dataflows are types of supervised ML models, which means that they learn from the known outcomes of past observations to predict the outcomes of other new unseen observations. The input dataset for training an AutoML model is a set of records that are labeled with the known outcomes.

After an ML model is trained, AutoML automatically generates a Power BI report that explains the likely performance of your ML model. AutoML emphasizes explainability, by highlighting the key influencers among your inputs that influence the predictions returned by your model. The report also includes key metrics for the model, depending on the ML model type.

Other pages of the generated report show the statistical summary of the model and the training details. The statistical summary is of interest to users who would like to see the standard data science measures of performance for the model. The training details summarize all the iterations that were run to create your model, with the associated modeling parameters. It also describes how each input was used to create the ML model.

You can then apply your ML model to your data for scoring. When the dataflow is refreshed, the predictions from your ML model are automatically applied to your data. Power BI also includes an individualized explanation for each specific prediction score that the ML model produces.

Creating an AutoML Model Step by Step

1- Data prep for creating ML Model:

Create a dataflow for the data with the historical outcome information, which is used for training the ML model.

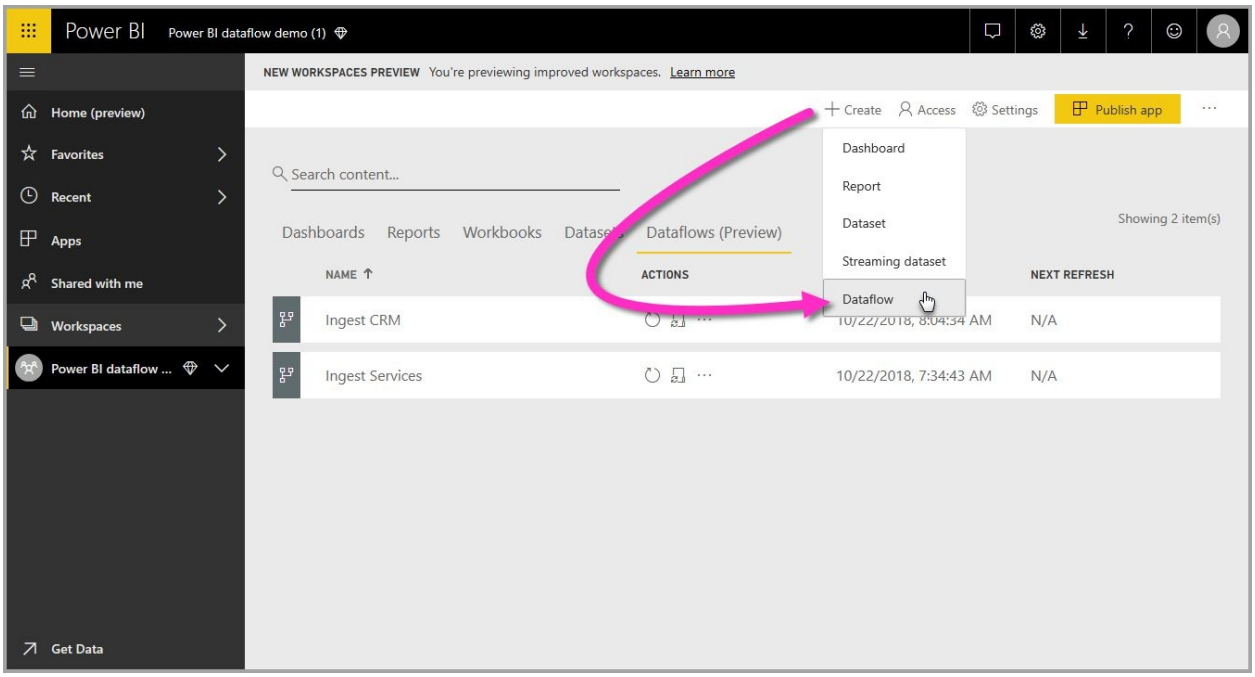


Figure 12-07: Create Dataflow

In the current release, Power BI uses data from only a single entity to train the ML model. So, if your historical data consists of multiple entities, you must manually join the data into a single dataflow entity. You should also add calculated columns for any business metrics that may be strong predictors for the outcome you're trying to predict.

2- Configuring the ML Model Inputs

AutoML has specific data requirements for training a ML model based on respective model types.

To create an AutoML model, select the ML icon in the " Actions" column of the dataflow entity with the historical data, and select "Add a ML model".

Figure 12-08: Add ML model

A simplified experience is launched, consisting of a wizard that guides you through the process of creating the ML model. The wizard includes the following simple steps.

1- Select the entity with the historical outcome data, and the field for which you want a prediction
2- Choose a model type based on the type of prediction you'd like to see
3- Select the inputs you want the model to use as predictive signals
4- Name your model and save your configuration

The historical outcome field identifies the label attribute for training the ML model, shown in the following image.

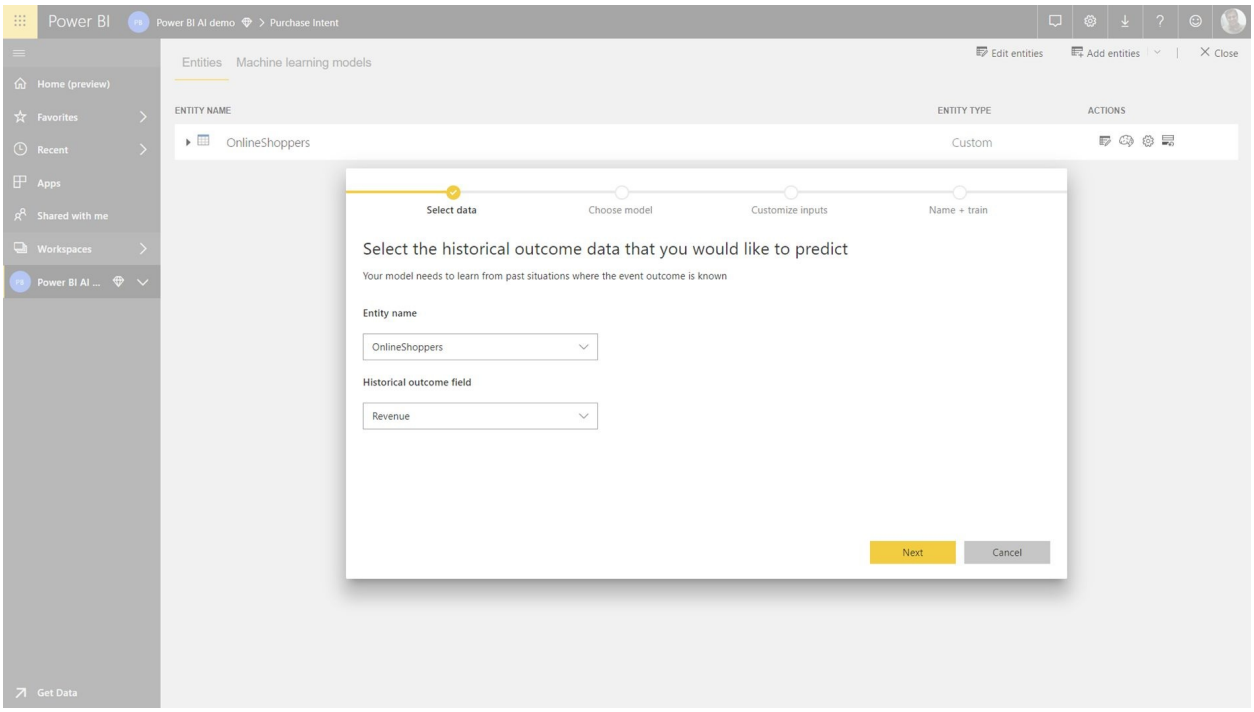


Figure 12-09: Select Outcome

When you specify the historical outcome field, AutoML analyzes the label data to identify the types of ML models that can be trained for that data and suggests the most likely ML model type that can be trained. Some model types may not be supported for the data that you have selected.

AutoML also analyzes all the fields in the selected entity to suggest the inputs that can be used for training the ML model. This process is approximate and is based on statistical analysis, so you should review the inputs used. Any inputs that are directly dependent on the historical outcome field (or the label field) should not be used for training the ML model, since they will affect its performance.

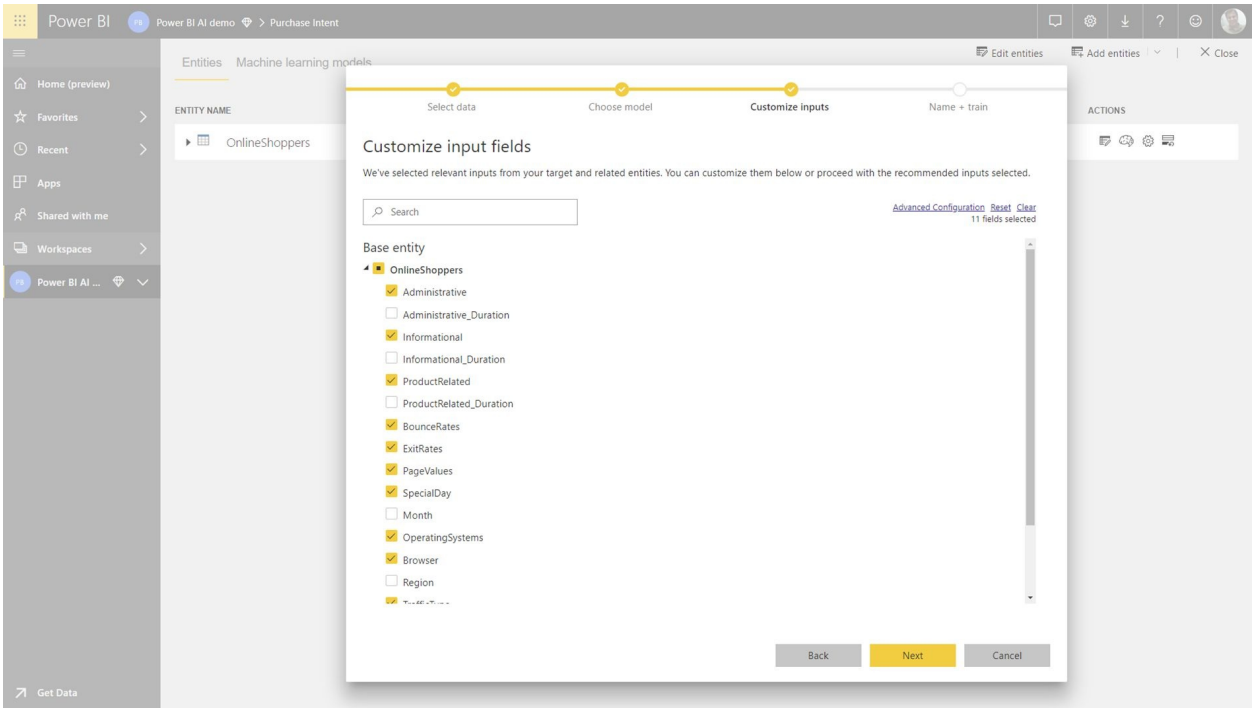


Figure 12-10: Select Inputs

In the final step, you can name the model and save its settings. At this stage, you are prompted to refresh the dataflow, which begins the training process for the ML model.

3- ML Model Training

Training of AutoML models is a part of the dataflow refresh. AutoML first prepares your data for training. AutoML splits the historical data you provide into a training and testing datasets. The test dataset is a holdout set that is used for validating the model performance after training. These are realized as Training and Testing entities in the dataflow. AutoML uses cross-validation for the model validation.

Next, each input field is analyzed and imputation is applied, which replaces any missing values with substituted values. A couple of different imputation strategies are used by AutoML. Then, any required sampling and normalization are applied to your data. AutoML applies several transformations at each selected input field based on its data type, and its statistical properties. AutoML uses these transformations to extract features for use in training your ML model.

The training process for AutoML models consists of up to 50 iterations with different modeling algorithms and hyperparameter settings to find the model with the best performance. The performance of each of these models is assessed

by validation with the holdout test dataset. During this training step, AutoML creates several pipelines for training and validation of these iterations. The process of assessing the performance of the models can take time, anywhere from several minutes to a couple of hours, depending on the size of your dataset and the dedicated capacity resources available.

In some cases, the final model generated may use ensemble learning, where multiple models are used to deliver better predictive performance.

4- AutoML Model Explainability

After the model has been trained, AutoML analyzes the relationship between the input features and the model output. It assesses the magnitude and direction of change to the model output for the holdout test dataset for each input feature. This is known as the *feature importance*.

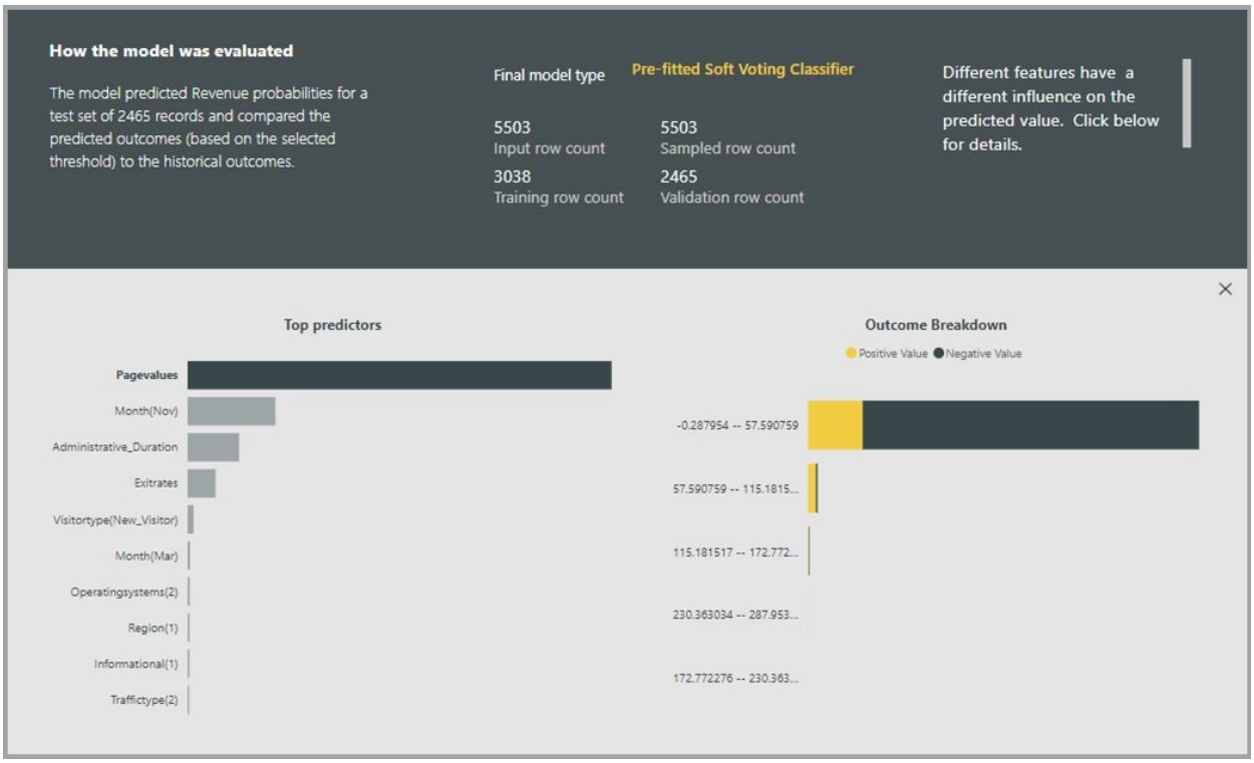


Figure 12-11: Model Report

5- AutoML Model Report

AutoML generates a Power BI report that summarizes the performance of the model during validation, along with the global feature importance. The report summarizes the results from applying the ML model to the holdout test data and comparing the predictions with the known outcome values.

You can review the model report to understand its performance. You can also

validate that the key influencers of the model align with the business insights about the known outcomes.

The charts and measures used to describe the model performance in the report depend on the model type. These performance charts and measures are described in the following sections.

Additional pages in the report may describe statistical measures about the model from a data science perspective. For instance, the **Binary Prediction** report includes a gain chart and the ROC curve for the model.

The reports also include a **Training Details** page that includes a description of how the model was trained, and includes a chart describing the model performance over each of the iterations runs.

Figure 12-12: Model Performance

Another section on this page describes how the imputation method used for filling missing values for the input fields, as well as how each input field was transformed to extract the features used in the model. It also includes the parameters used by the final model.

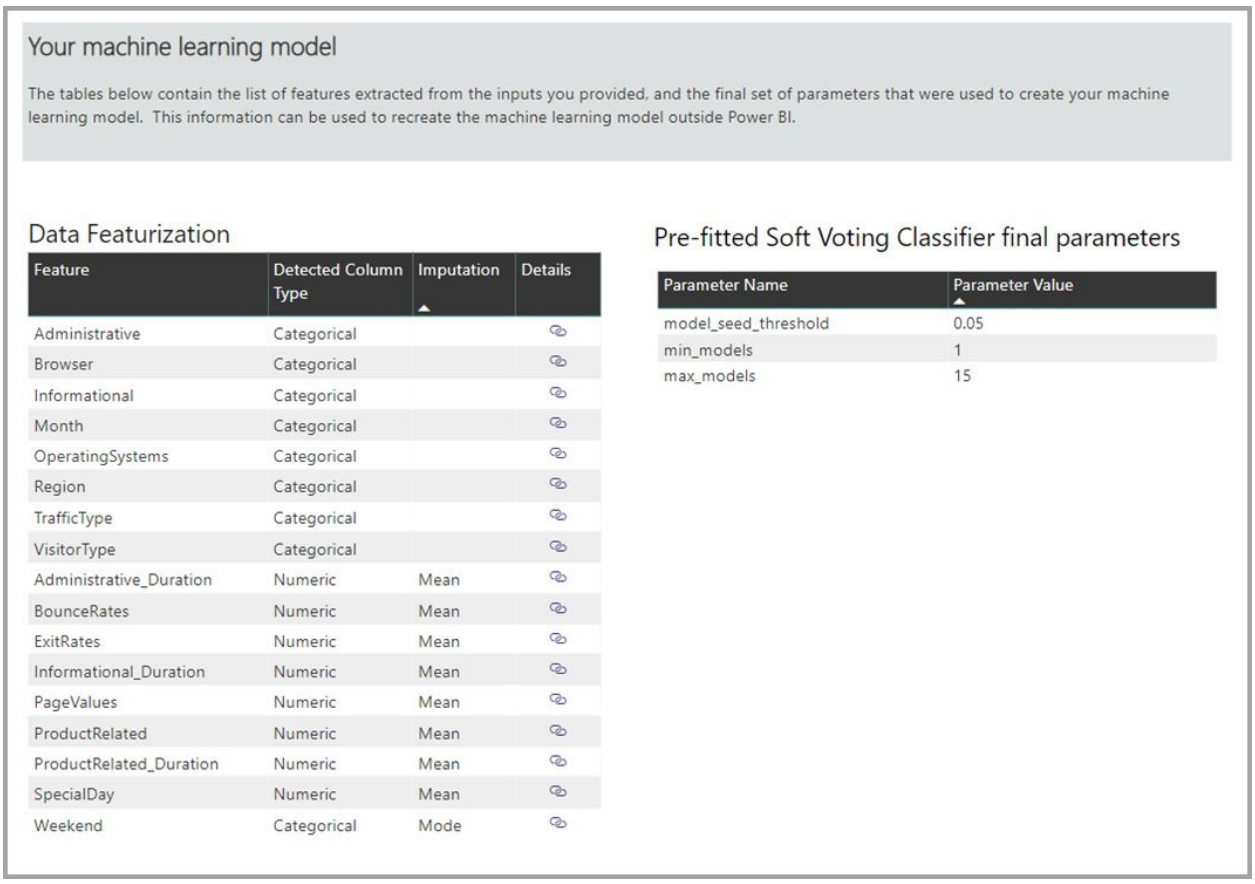


Figure 12-13: Data Featurization and Parameters

If the model produced uses ensemble learning, then the Training Details page also includes a section describing the weight of each constituent model in the ensemble, as well as its parameters.

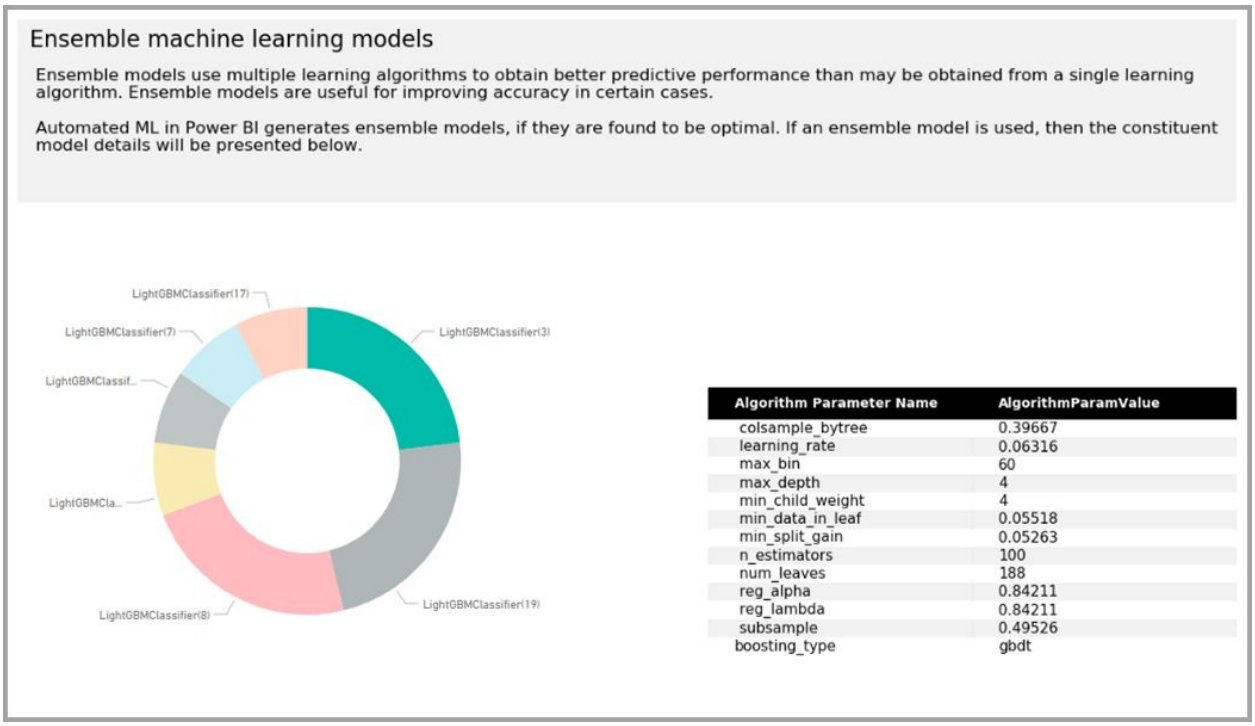


Figure 12-14: Ensemble ML

6- Applying the AutoML Model

If you're satisfied with the performance of the ML model created, you can apply it to new or updated data when your dataflow is refreshed. You can do this from the model report, by selecting the Apply button in the top-right corner.

To apply the ML model, you must specify the name of the entity to which it must be applied, and a prefix for the columns that will be added to this entity for the model output. The default prefix for the column names is the model name. The *Apply* function may include additional parameters specific to the model type.

Applying the ML model creates a new dataflow entity with the suffix *enriched <model_name>*. For instance, if you apply the *PurchaseIntent* model to the *OnlineShoppers* entity, the output will generate the *OnlineShoppers enriched PurchaseIntent.*

Currently, the output entity cannot be used to preview the ML model results in the Power Query editor. The output columns always show null as the result. To view the results, a second output entity with the suffix *enriched <model_name> Preview* is created when the model is applied.

You must refresh the dataflow, to preview the results in the Query Editor.

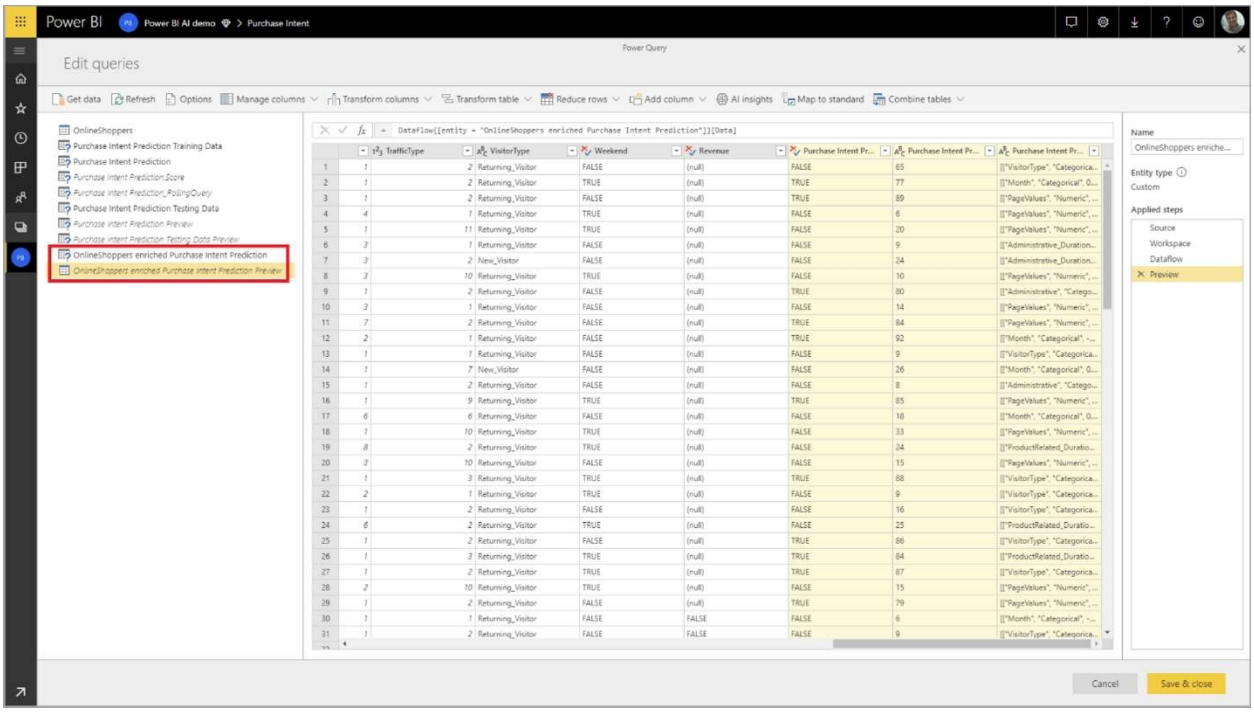


Figure 12-15: Output Entity

When you apply the model, AutoML always keeps your predictions up-to-date when the dataflow is refreshed.

AutoML also includes an individualized explanation for each row that it scores in the output entity.

To use the insights and predictions from the ML model in a Power BI report, you can connect to the output entity from Power BI Desktop using the dataflows connector.

Deep dive into the 3 types of ML Models

1- Binary Prediction Models

Binary Prediction models, more formally known as binary classification models, are used to classify a dataset into two groups. They're used to predict events that can have a binary outcome, such as whether a sales opportunity will convert, whether an account will churn, whether an invoice will be paid on time; whether a transaction is fraudulent, and so on.

Since the outcome is binary, Power BI expects the label for a binary prediction model to be a Boolean, with known outcomes being labeled true or false. For instance, in a sales opportunity conversion model, sales opportunities that have been won are labeled true, those that have been lost are labeled false, and the open sales opportunities are labeled null.

The output of a Binary Prediction model is a probability score, which identifies the likelihood that the outcome corresponding to the label value being true will be achieved.

Training a Binary Prediction Model

To create a Binary Prediction model, the input entity containing your training data must have a Boolean field as the historical outcome field to identify the past known outcomes.

Pre-requisites:

- A Boolean field must be used as the historical outcome field
- A minimum of 50 rows of historical data is required for each class of outcomes

In general, if the past outcomes are identified by fields of a different data type, you can add a calculated column to transform these into a Boolean using Power Query.

The process of creation for a Binary Prediction model follows the same steps as other AutoML models, described in the section Configuring the ML model inputs above.

Binary Prediction Model Report

The Binary Prediction model produces as an output a probability that a record will achieve the outcome defined by the Boolean label value as True. The report includes a slicer for the probability threshold, which influences how the scores above and below the probability threshold are interpreted.

The report describes the performance of the model in terms of *True Positives*, *False Positives*, *True Negatives* and *False Negatives*. True Positives and True Negatives are correctly predicted outcomes for the two classes in the outcome data. False Positives are outcomes that had the actual Boolean label of value False but were predicted as True. Conversely, False Negatives are outcomes where the actual Boolean label value was True but were predicted as False.

Measures, such as Precision and Recall, describe the effect of the probability threshold on the predicted outcomes. You can use the probability threshold slicer to select a threshold that achieves a balanced compromise between Precision and Recall.

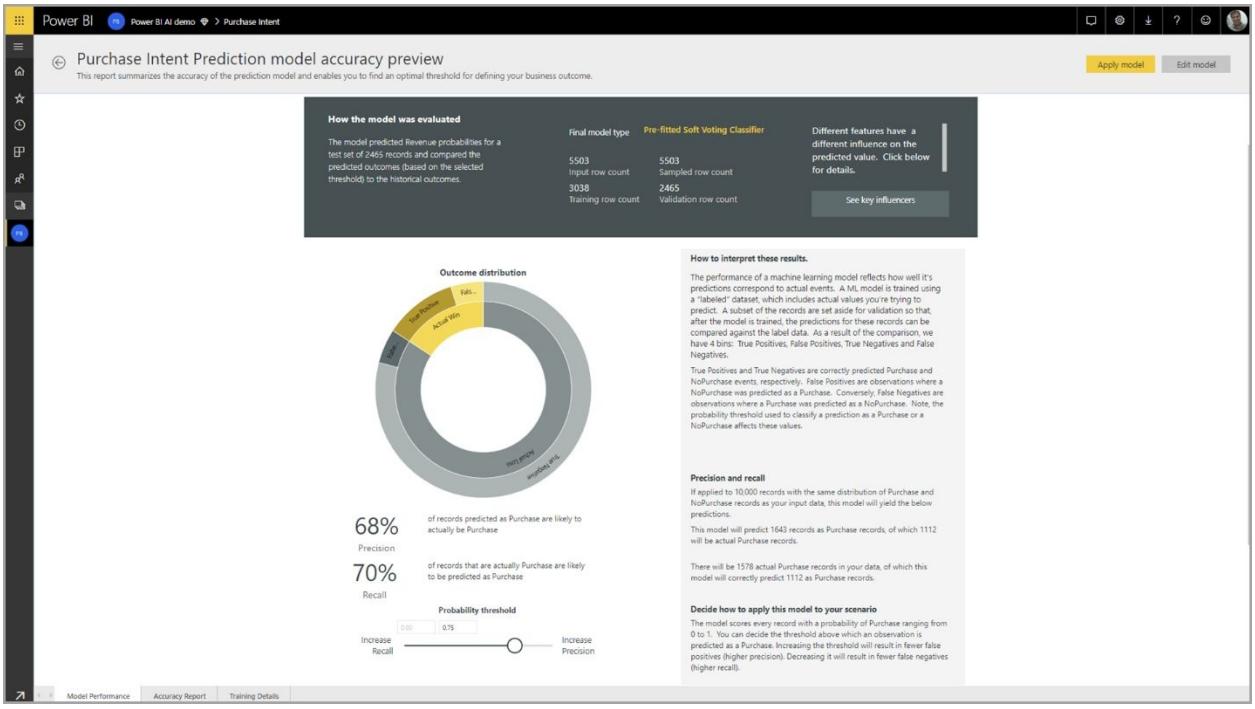


Figure 12-16: Binary Prediction Report

The Accuracy Report page of the model report includes the *Cumulative Gains* chart and the ROC curve for the model. These are statistical measures of model performance. The reports include descriptions of the charts shown.

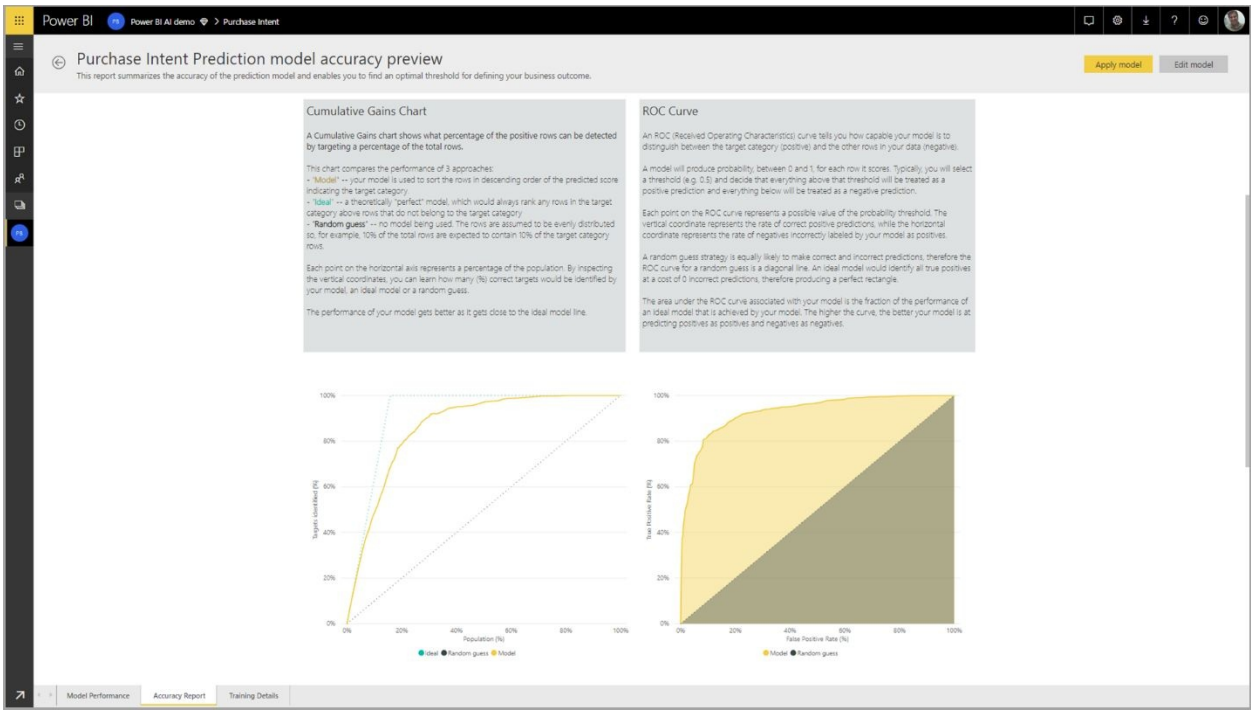


Figure 12-17: Binary Prediction Model Accuracy

Applying a Binary Prediction Model

To apply a Binary Prediction model, you must specify the entity with the data to which you want to apply the predictions from the ML model. Other parameters include the output column name prefix and the probability threshold for classifying the predicted outcome.

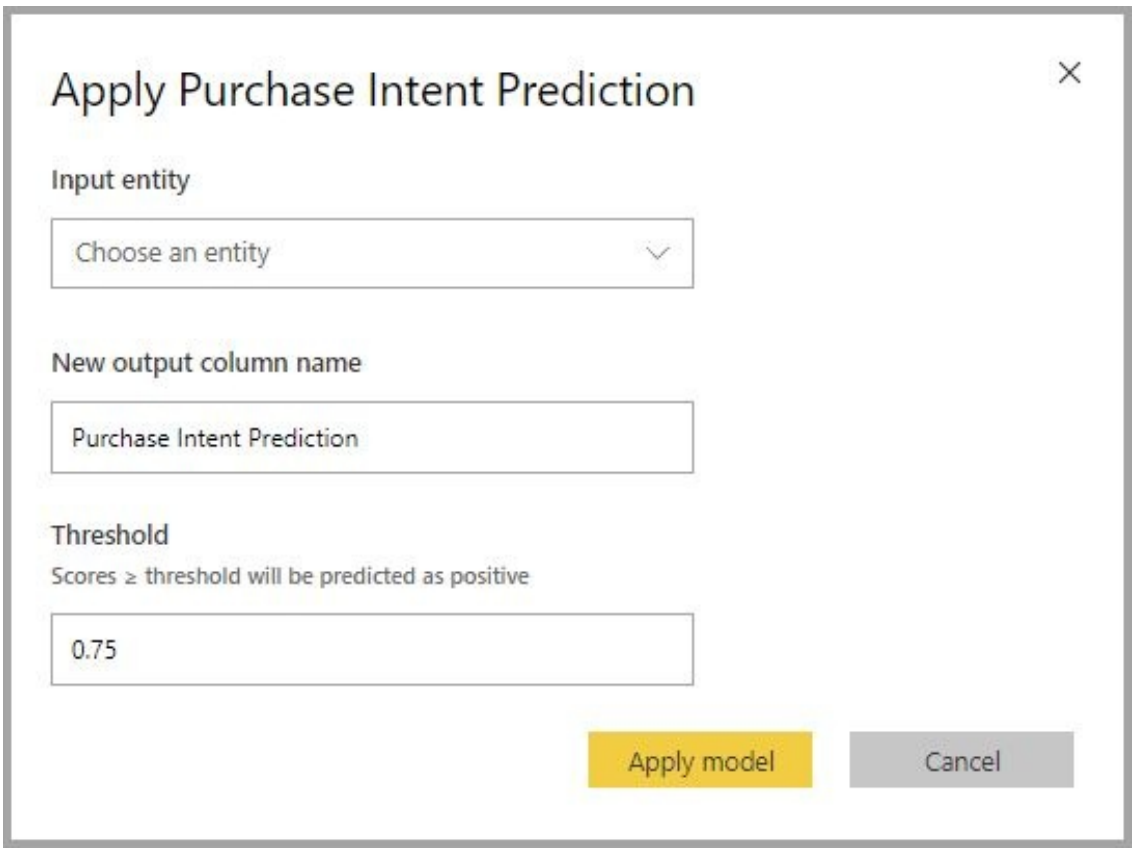


Figure 12-18: Applying Binary Prediction Model

When a Binary Prediction model is applied, it adds three output columns to the enriched output entity. These are the *PredictionScore*, *PredictionOutcome* and *PredictionExplanation*. The column names in the entity have the prefix specified when the model is applied.

The *PredictionOutcome* column contains the predicted outcome label. Records with probabilities exceeding the threshold are predicted as likely to achieve the outcome, and those below are predicted as unlikely to achieve the outcome.

The *PredictionExplanation* column contains an explanation with the specific influence that the input features had on the *PredictionScore*. This is a JSON formatted collection of weights of the input features for the prediction.

2- Classification Models

Classification models are used to classify a dataset into multiple groups or classes. They are used to predict events that can have one of multiple possible outcomes, such as whether a customer is likely to have a very high, high, medium, or low Lifetime Value; whether the risk for default is High, Moderate, Low or Very Low; and so on. The output of a Classification model is a

probability score, which identifies the likelihood that a record will achieve the criteria for a given class.

Training a Classification Model

The input entity containing your training data for a Classification model must have a string or numeric field as the historical outcome field, which identifies the past known outcomes.

Pre-requisites:

- A minimum of 50 rows of historical data is required for each class of outcomes

The process of creation for a Classification model follows the same steps as other AutoML models, described in the section Configuring the ML model inputs above.

Classification Model Report

The Classification model report is produced by applying the ML model to the holdout test data and comparing the predicted class for a record with the actual known class.

The model report includes a chart that includes the breakdown of the correctly and incorrectly classified records for each known class.

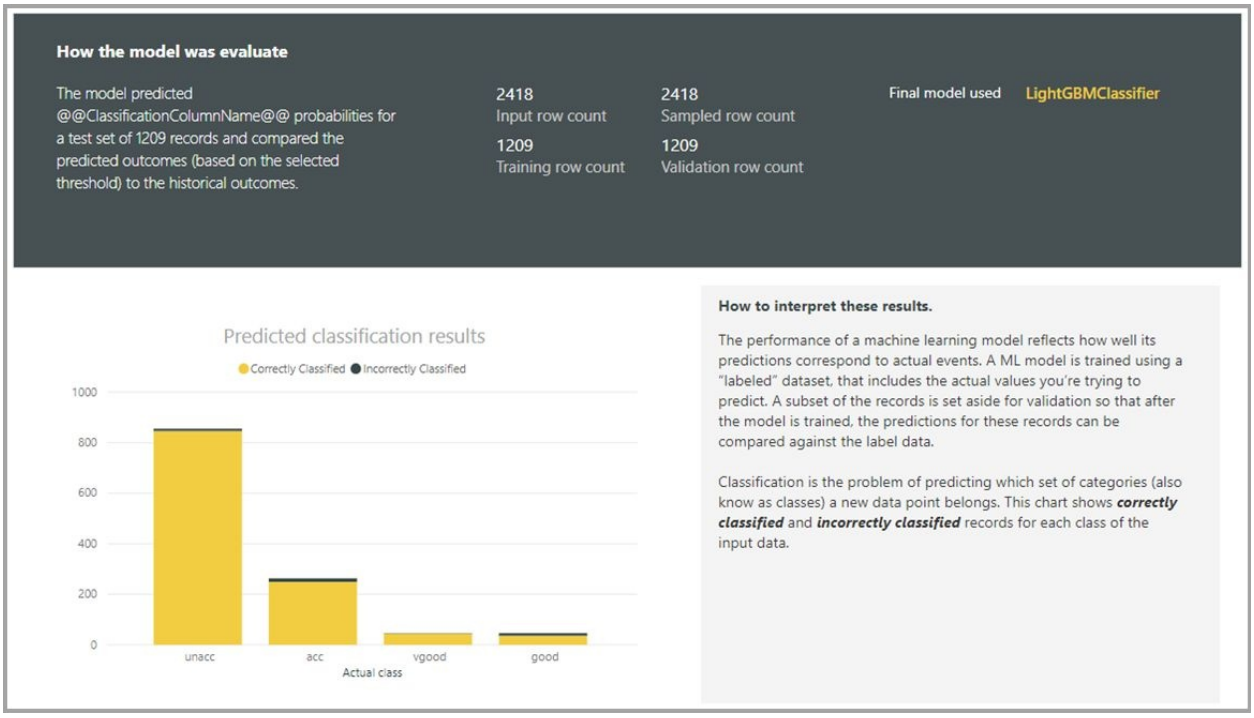


Figure 12-19: Classification Model Report

A further class-specific drilldown enables an analysis of how the predictions for a known class are distributed. This includes the other classes in which records of that known class are likely to be misclassified.

Figure 12-20: Classification Model Accuracy

The model explanation in the report also includes the top predictors for each class.

The Classification model report also includes a Training Details page similar to the pages for other model types, as described in the section AutoML model report earlier in this article.

Applying a classification model

To apply a Classification ML model, you must specify the entity with the input data and the output column name prefix.

When a Classification model is applied, it adds three output columns to the enriched output entity. These are the *PredictionScore*, *PredictionClass* and *PredictionExplanation*. The column names in the entity have the prefix specified when the model is applied.

The *PredictionClass* column contains the most likely predicted class for the record. The *PredictionScore* column contains the list of probability scores for the record for each possible class.

The *PredictionExplanation* column contains an explanation with the specific influence that the input features had on the *PredictionScore*. This is a JSON formatted collection of weights of the input features for the prediction.

3- Regression Models

Regression models are used to predict a value, such as the revenue likely to be realized from a sales deal, the lifetime value of an account, the amount of a receivable invoice that is likely to be paid, the date on which an invoice may be paid, and so on. The output of a Regression model is the predicted value.

Training a Regression Model

The input entity containing the training data for a Regression model must have a numeric field as the historical outcome field, which identifies the past known outcome values.

Pre-requisites:

- A minimum of 100 rows of historical data is required for a Regression model

The process of creation for a Regression model follows the same steps as other AutoML models, described in the section Configuring the ML model inputs above.

Regression Model Report

Like the other AutoML model reports, the Regression report is based on the results from applying the model to the holdout test data.

The model report includes a chart that compares the predicted values to the actual value. In this chart, the distance from the diagonal indicates the error in the prediction.

The residual error chart shows the distribution of the percentage of average error for different values in the holdout test dataset. The horizontal axis represents the mean of the actual value for the group, with the size of the bubble showing the frequency or count of values in that range. The vertical axis is the average residual error.

Figure 12-21: Regression Model Performance

The Regression model report also includes a Training Details page like the reports for other model types, as described in the section AutoML model report above.

Applying a Regression Model

To apply a Regression ML model, you must specify the entity with the input data and the output column name prefix.

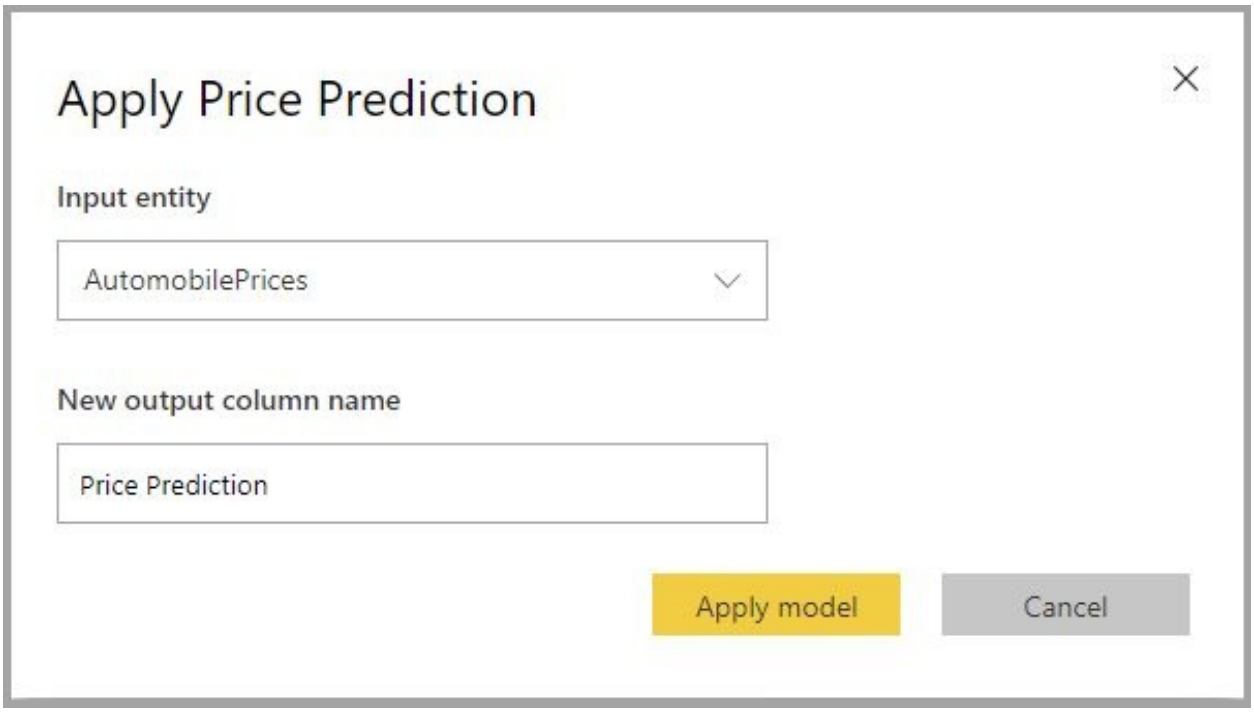


Figure 12-22: Apply Regression Model

When a Regression model is applied, it adds two output columns to the enriched output entity. These are the *PredictionValue*, and *PredictionExplanation*. The column names in the entity have the prefix specified when the model is applied.

The *PredictionValue* column contains the predicted value for the record based on the input fields. The *PredictionExplanation* column contains an explanation with the specific influence that the input features had on the *PredictionValue*. This is a JSON formatted collection of weights of the input features.

Summary

Automated machine learning, also referred to as AutoML, is the process of automating the time consuming, iterative tasks of ML model development. It allows data scientists, analysts, and developers to build ML models with high scale, efficiency, and productivity all while sustaining model quality.

Traditional ML model development is resource-intensive, requiring significant domain knowledge and time to produce and compare dozens of models. AutoML in Power BI empowers Citizen Data Scientists to accelerate the time it takes to get production-ready ML models with great ease and efficiency.

AutoML is available for dataflows in workspaces hosted on Power BI Premium and Embedded capacities only. If you don’t have those options, you still can do Machine Learning with Power BI using Python/R, Integration with Azure Machine Learning Studio or Integration with Azure Machine Learning Services.

About the Author

Ashraf Ghonaim, Strategic Management Consultant at the City of Toronto. He holds a Computer Engineering degree and an MBA degree in Strategic Management with special emphasis on IT-Business Strategic Alignment using Balanced Scorecard. He is also a certified Balanced Scorecard Professional™ from the co-creators Drs. Kaplan and Norton, a Lean Six Sigma Black Belt and a Project Management Professional (PMP). Ashraf's areas of expertise are; Strategy Management, Performance Measurement, Process Improvement, Analytics and Visualization.

Ashraf is the leader of the Power Platform user group community in Toronto that has more than 1,400 active members. He is also the organizer of and speaker at the Power Platform World Tour events in Toronto and Montreal and also a frequent speaker at other Microsoft's events like Global AI Bootcamp, Azure Bootcamp, D365, SQL Saturdays, Sharepoint Saturdays, etc.

Ashraf also volunteers in several Open Data and Data for Public Good initiatives. Ashraf is very active in participating in datathons and hackathons as a mentor and a judge. He recently got his Microsoft MVP award in recognition of his contribution to the data and analytics community. Ashraf is an MVP in Data Platform (Power BI) with special focus on leveraging Machine Learning, AI Cognitive Services and Advanced Analytics Capabilities in Power BI to deliver impactful results.

Part V: Integration of Other Applications with Power BI

Chapter 13: Power BI REST API

Author: Eduardo Castro

Power BI service components can be configured and administered using its user interface on the web available at http://app.powerbi.com, however sometimes automation is needed in order to perform repetitive tasks or integration.

In this chapter we will explore how to use the Power BI REST API to administer and integrate Power BI with other applications. We will use C# to manage workspaces, permissions and other administration related task using Power BI REST API.

Getting ready to use Power BI REST API

The Power BI REST API is available to developers to take advantage of when you need to perform repetitive administration, including configuring and integrating Power BI with external applications. However, before you can use the REST API you need to have the proper permissions and configuration.

The first step is to register your application. This is a requirement because any application that wants to interact with the Power BI Service and with Azure must be registered and authorized in the Power BI and Azure tenants.

Register your developer application

To start the registration process, navigate to https://dev.powerbi.com/apps.In this first step you must first login using your corporate Power BI account and must follow the instructions to register and authorize your application.

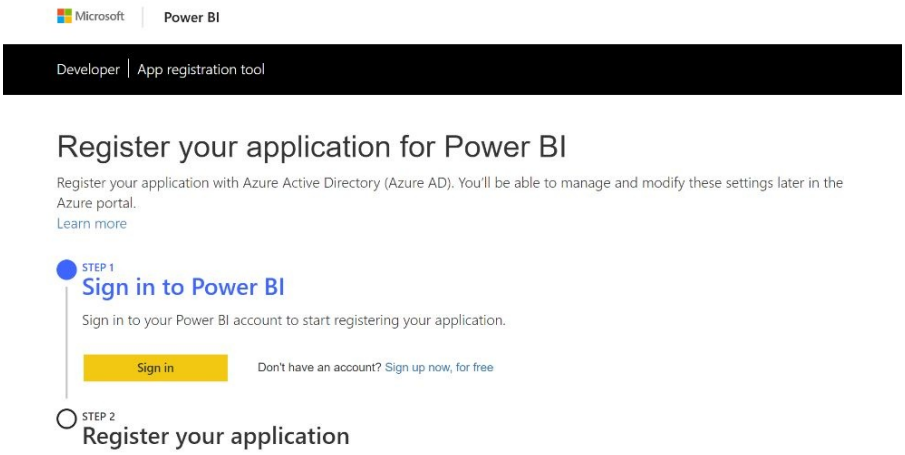


Figure 13-01: Beginning the Power BI Application Registration

Click on the “Sign In” button and login with your credentials.

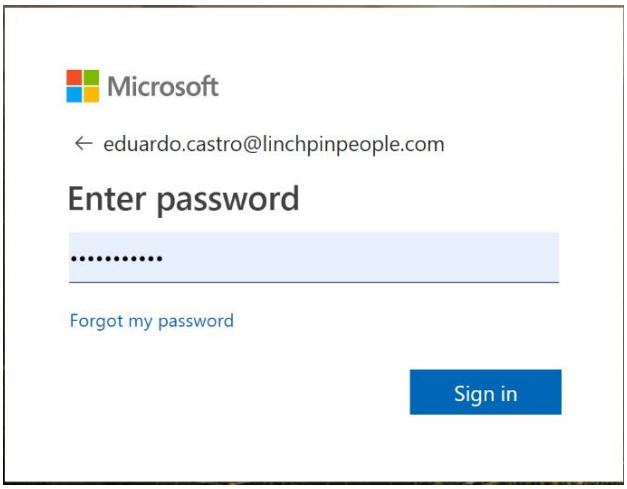


Figure 13-02: Logging with your credentials

After logging in you can continue with your app registration.

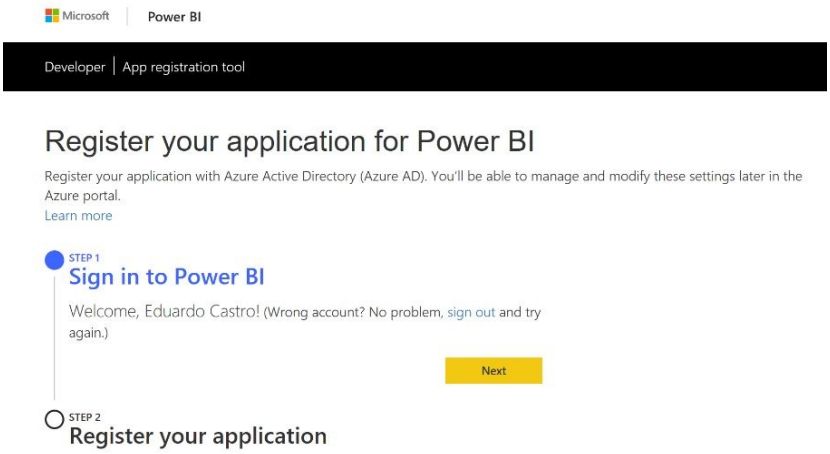


Figure 13-03: Begin App registration

The App Registration page requires the following information: your application name and the application type. For the application type you have two options:

- Server-side Web App: This use case is for applications that resides on a server. In this case it can be web applications or mobile applications.
- Native App: This use case is for applications that will run on a specific environment, for example a console application.

During the App Registration you must provide your application name and application URL and redirect URL.

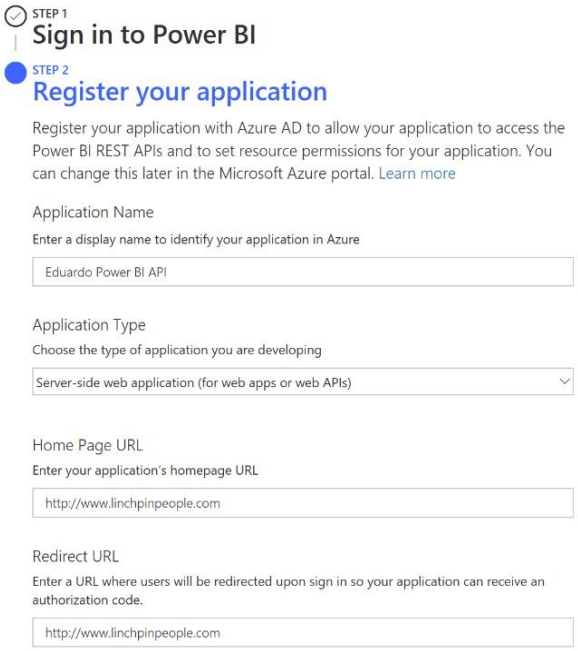


Figure 13-04: Basic app information

The application type is related with the integration of Power BI with your applications, this is because there are two ways to integrate Power BI into your application:

- Integration with a token: In this case your application is the one that authenticates with the Power BI Service, all the authentication process is done by the application, and end users do not even need a Power BI Account.
- Integration without a token: In this case, when the users want to access your application content, they will be prompted for a Power BI Account for authentication.

If your application is going to use integration with a token you must create the application as Native Application Type, it doesn't matter is the application you are creating is web-based, console or a mobile application.

The next step is to choose the proper permissions you want for your application, you must choose only the required permissions based on the operations you plan to use with the Power BI API. Your application can have access only to the "Read Only APIs" this is useful when your are creating an application that is going to use to generate reports about your Power BI Objects, but if your application needs to modify the content then you will need to give your application "Read and write APIs" permissions.

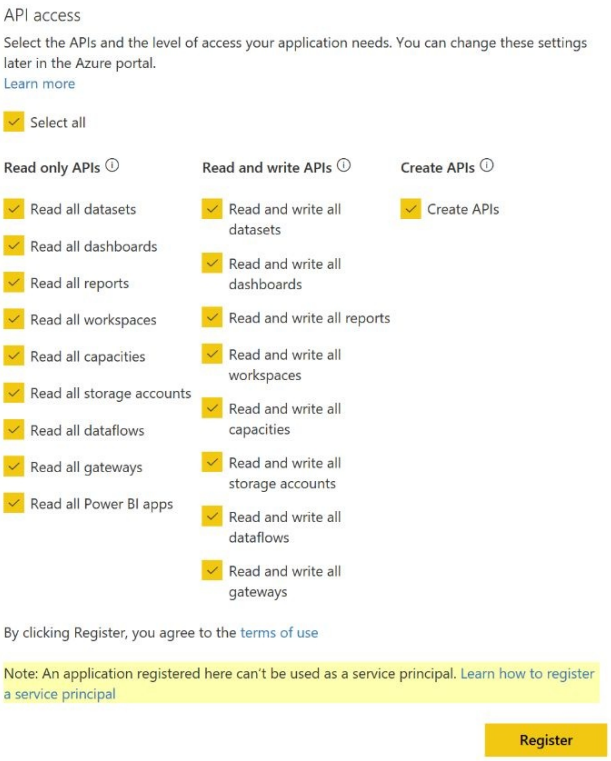


Figure 13-05: Choosing the API access permissions

The last step is the register your application, once you have selected all the properties and permissions of your applications, you click the “Register” button and if everything is ok, you will receive a Client Secret and Client ID, copy & paste this information and save it, because you will need it in your code in order to interact with Power BI Service.

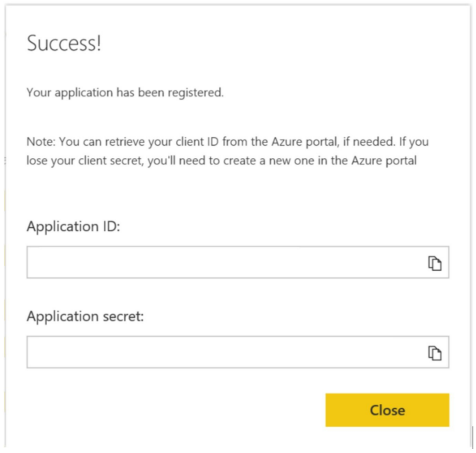


Figure 13-06: Power BI Application successfully registered

Register your application in Azure Portal

Another method to register your application is using the Azure Portal, to use this method you must go to http://portal.azure.com using this method gives your more control on your application and if you want you can always go to the Azure Portal and change the permissions of your application.

The first step is to login in http://portal.azure.com, once you are authenticated in the Portal you must select the Azure Active Directory Configuration, as shown below.

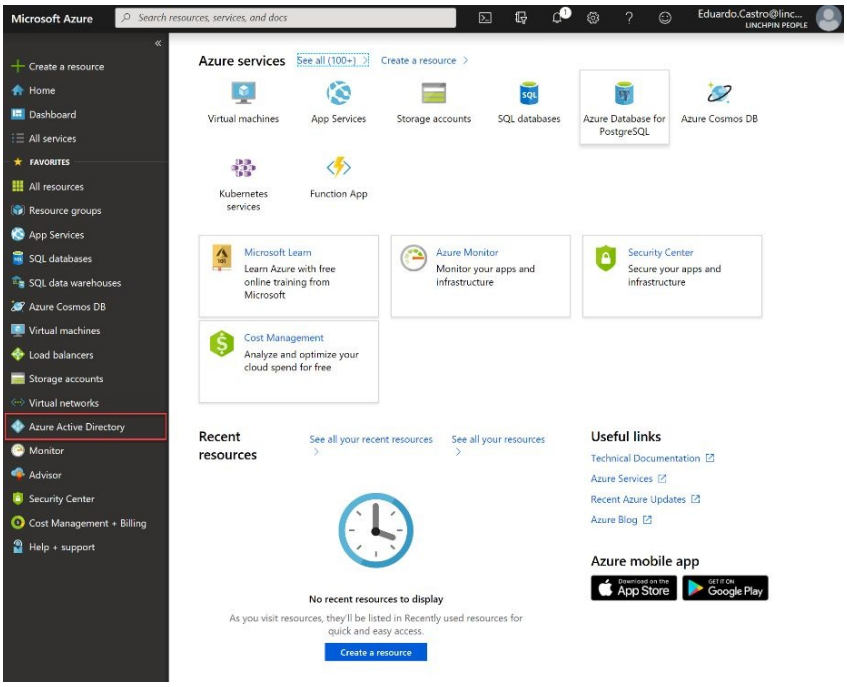


Figure 13-07: Power BI Application registration using the Azure Portal

The next step is to choose the App Registration Option.

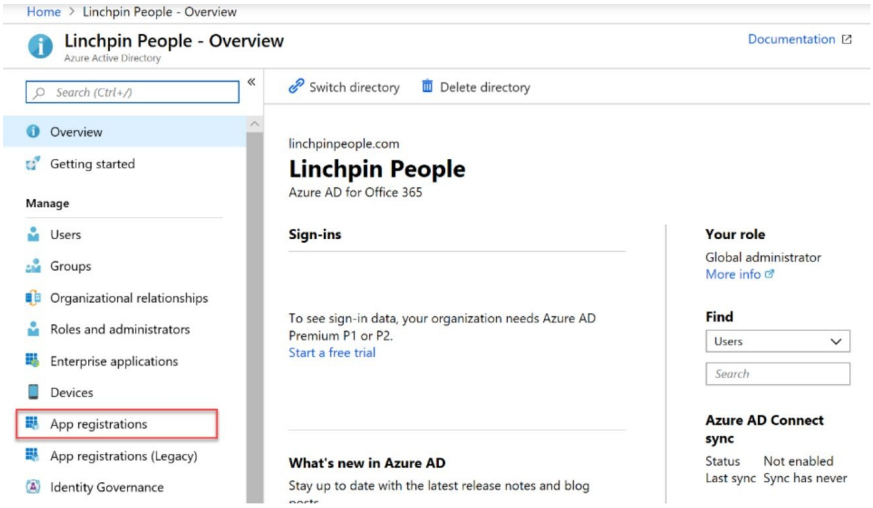


Figure 13-08: Application registration using the Azure Portal

In the App registrations page, you will see all your previously created applications and you can create new ones. In our case, we will create a new application registration.

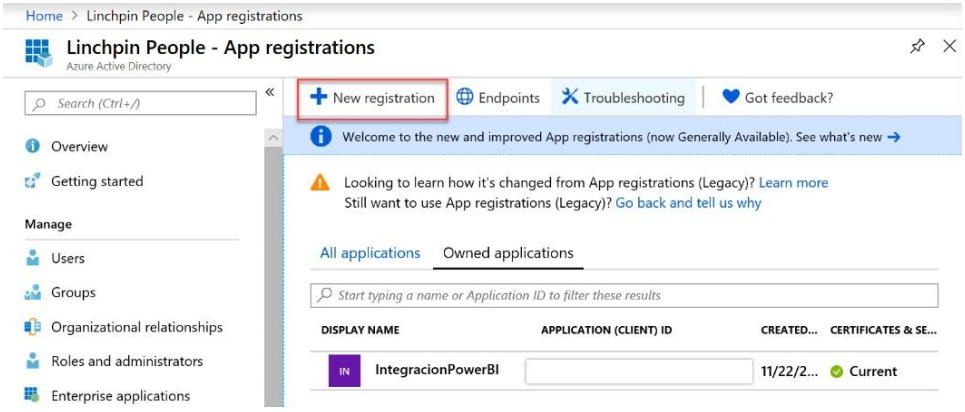


Figure 13-09: New application registration using the Azure Portal

Once you click New Registration, you must assign a name to your application and must choose one of the supported account types related to who can use this application or access this API.

- Accounts in this organizational directory only (Linchpin People). This option is the default if you are creating a line-of-business (LOB) application. This option maps your application to a single-tenant Azure AD.
- Accounts in any organizational directory
- Accounts in any organizational directory and personal Microsoft accounts (e.g. Skype, Xbox, Outlook.com)

If you want to restrict the use of the Application to only your internal users you must choose the first option. Once created you must assign the proper permissions to your application, to do so, click on API Permissions.

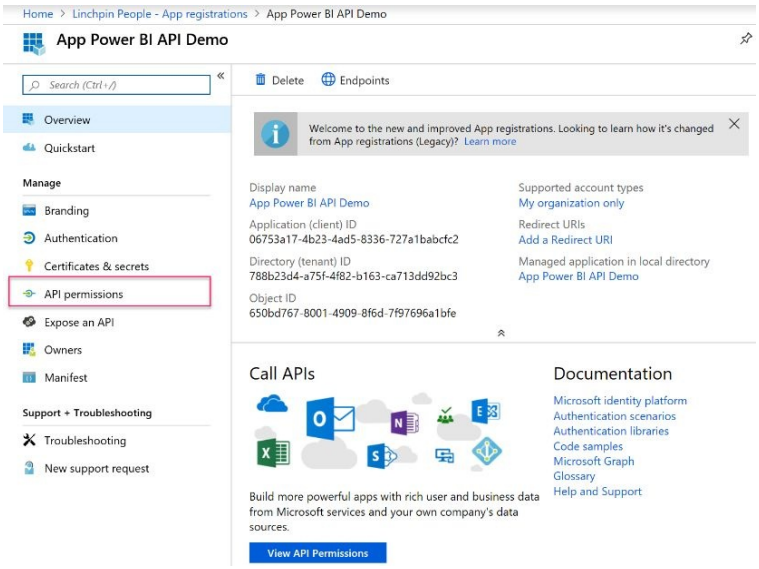


Figure 13-10: New application API Permissions

In the Request API permissions page, you must select Power BI Service to get the permission configuration related to Power BI.

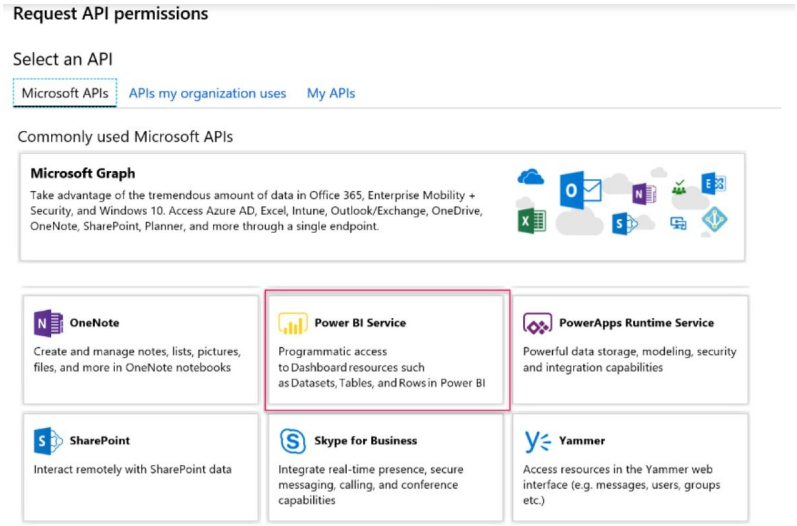


Figure 13-11: Power BI Service selection in permissions page

In the permission page you can assign or delegate your application one or more permission depending on the operations and interaction that your application will be doing using the REST API, in our case we will select all the permissions, as shown below.

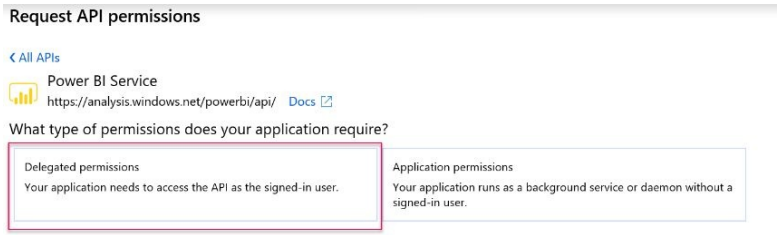


Figure 13-12: Power BI Service selection in permissions delegation

In the Delegated permissions page, you must select the permissions for your application and the click on Add permissions.

Figure 13-12: Power BI Service adding permissions

After you have added permissions a warning message is shown indicating that a tenant administrator must grant consent the permissions for your application.

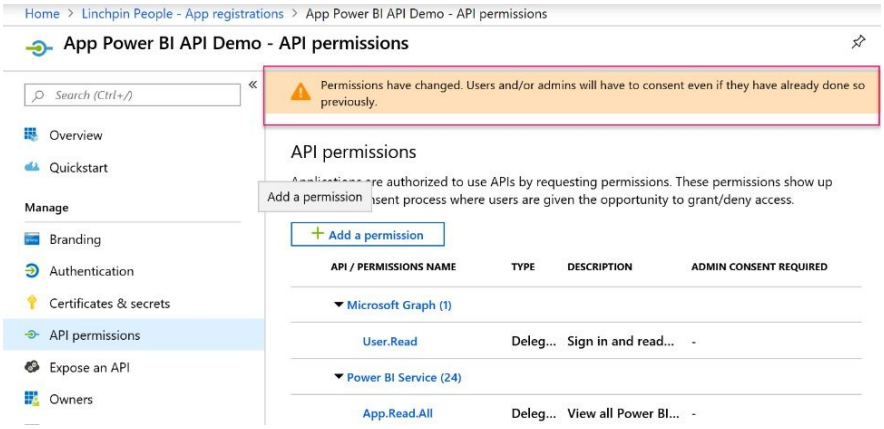


Figure 13-13: Power BI Service warns that administrator consent is required

If your are a tenant administrator you can grant consent yourself if not you will have to ask your tenant administrator to grant permissions consent to your application, as show below.

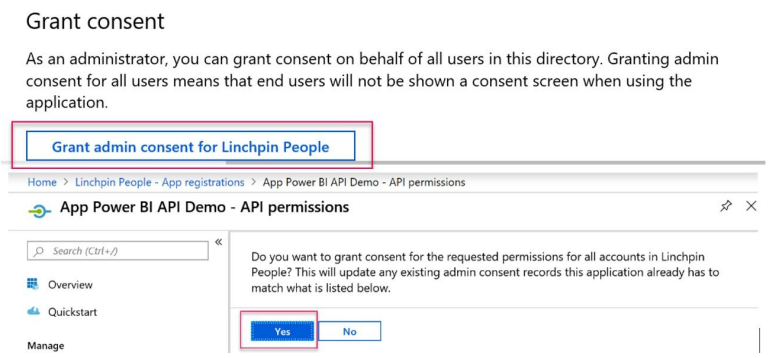


Figure 13-14: Administrator consent for the required permissions

After the permissions are granted the page will all the permissions that the application can use.

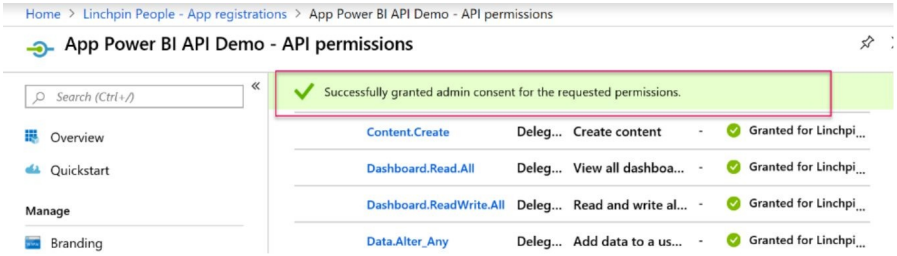


Figure 13-15: Administrator has granted consent for the required permissions

Before you can use the Power BI Rest API you must get the application secret. To do it click on “Certificates & secrets”

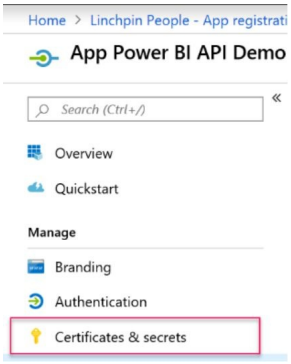


Figure 13-16: The application secret must be created before using the REST API

The next step is to create the Client Secret to be used in our code, select “Certificates & secrets” and then “New client secret”.

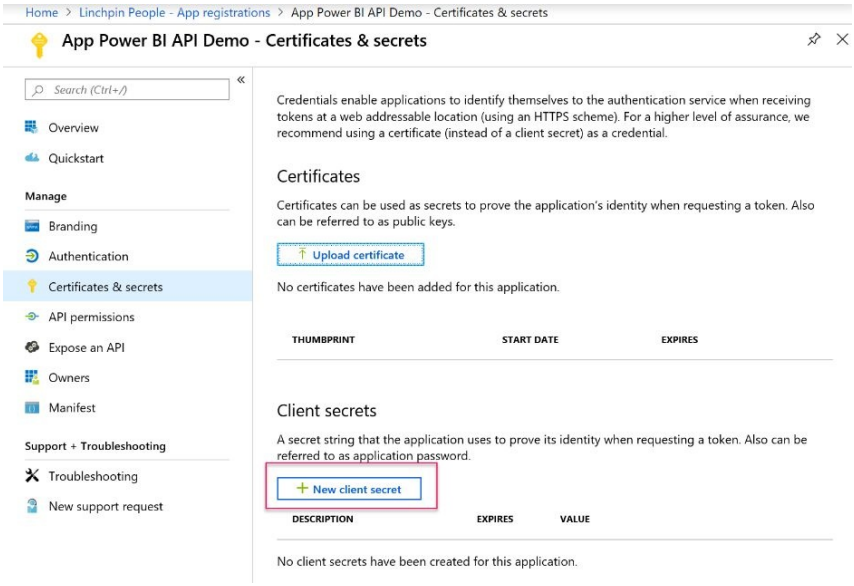


Figure 13-17: Creating of the client secret to be used with the REST API

In the client secret creation page, you must select the duration of the client secret, it can be 1 year, 2 years or without expiration.

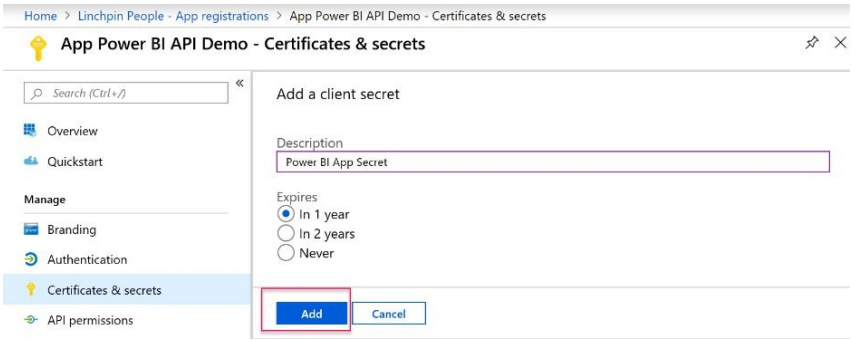


Figure 13-18: Client secret expiration configuration

After the client secret is created, it will be displayed next to the description field, be sure to write down the client secret because it will not be show after you navigate off this web page.

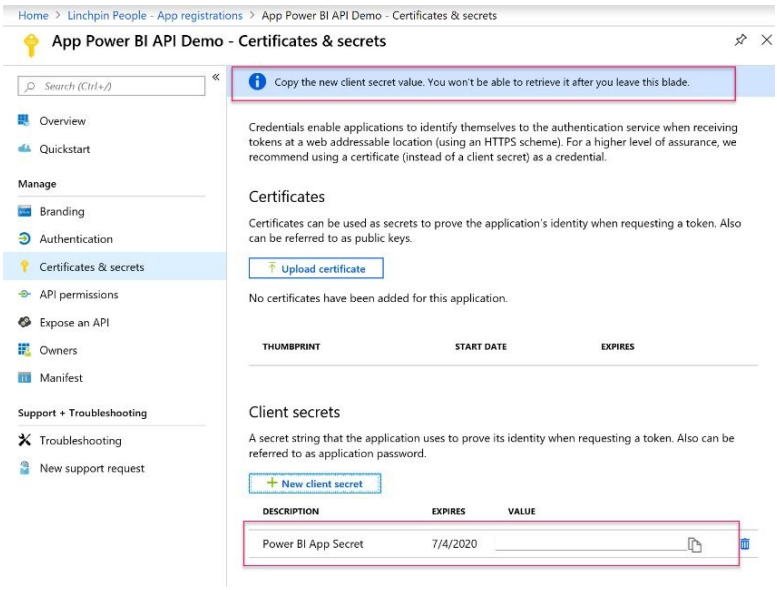


Figure 13-19: Client secret was created successfully

The final step is to configure our application to not use redirect URI, to do it go to Authentication section in the application configuration.

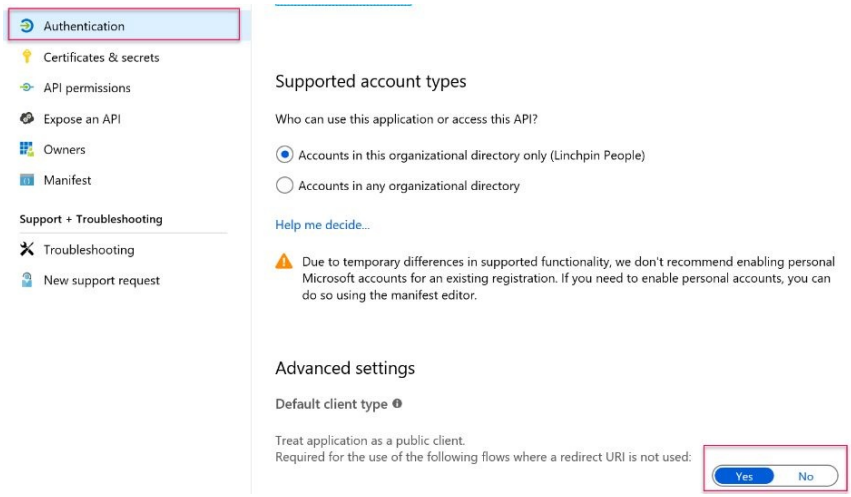


Figure 13-20: Configure your application to not use redirect URI

Now you are ready to start creating your C# application and to use the Power BI API.

Preparing Visual Studio to use the Power BI REST API

In this chapter we will use the sample code developed by Microsoft and that is available in GitHub at https://github.com/Microsoft/PowerBI-Developer-Samples/tree/master/App%20Owns%20Data, the code provided by Microsoft is free to use, and it will be base for our sample here.

Start by downloading or cloning the GitHub repository to your local machine, in our case we will download it.

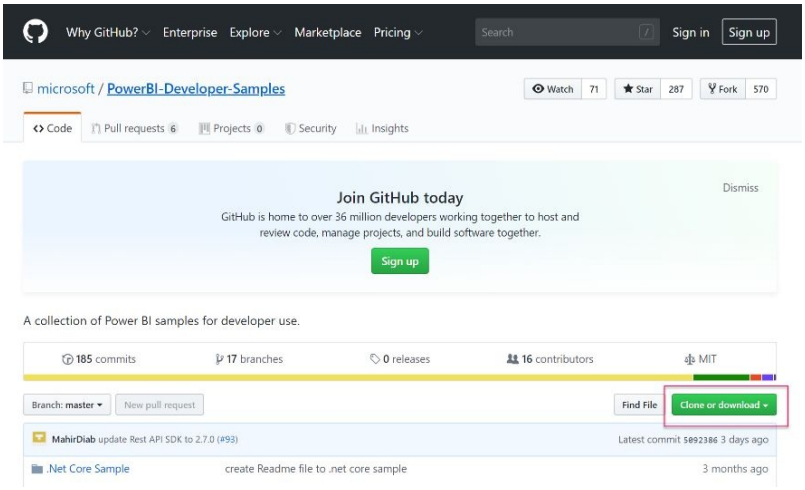


Figure 13-21: Download the sample code to your computer

Open the PowerBIEmbedded_AppOwnsData solution, this will the base code we will be using.

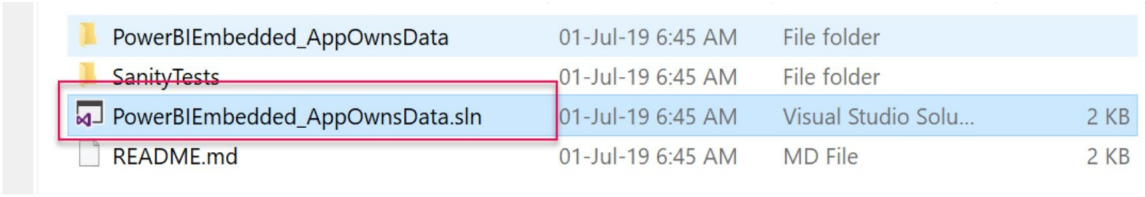


Figure 13-22: Sample code for using Power BI REST API

Before your application can use the Power BI REST API you must provide the proper configuration in the web.config file, the data you must fill are: applicationId, workspaceId, pbiUsername, pbiPassword, applicationSecret and tenant

```
  <add key="AuthenticationType" value="MasterUser"/>
  <!-- Common configuration properties for both authentication types -->
  <add key="applicationId" value=""/>
  <add key="workspaceId" value=""/>
  <!-- The id of the report to embed. If empty, will use the first report in group --
  <add key="reportId" value=""/>

  <!-- Fill Tenant ID in authorityUrl-->
  <add key="authorityUrl" value="https://login.microsoftonline.com/common/"/>
  <add key="resourceUrl" value="https://analysis.windows.net/powerbi/api"/>
  <add key="apiUrl" value="https://api.powerbi.com/"/>
  <add key="embedUrlBase" value="https://app.powerbi.com/"/>
</appSettings>

<MasterUser>
  <!-- Note: Do NOT leave your credentials on code. Save them in secure place. -->
  <add key="pbiUsername" value=""/>
  <add key="pbiPassword" value=""/>
</MasterUser>

<ServicePrincipal>
  <!-- Note: Do NOT leave your app secret on code. Save it in secure place. -->
  <add key="applicationSecret" value=""/>
  <add key="tenant" value=""/>
</ServicePrincipal>
```

Figure 13-23: Web.config configuration to use Power BI REST API

To get these values please get back to the Azure Portal and click on the application we had just created, the information we need will be displayed.

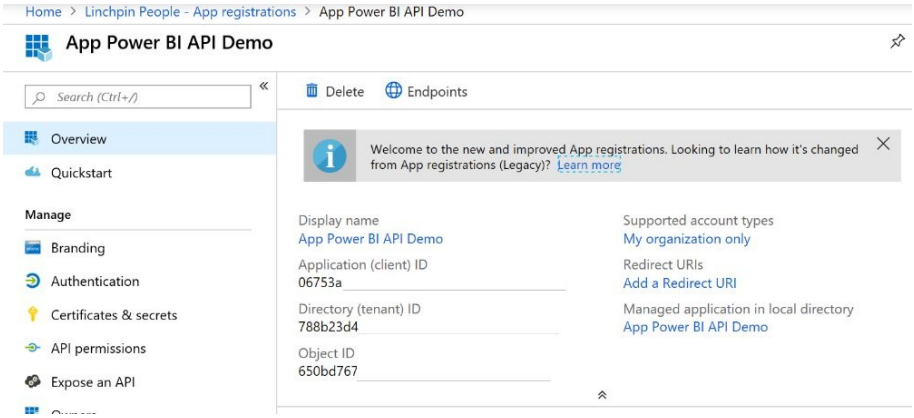


Figure 13-24: Information needed to be included in the web.config of your application

The following code invokes the Authentication Method and as result you will get valid token credentials that will be used in the following methods to invoke the REST API.

```
// Get token credentials for user
var getCredentialsResult = await GetTokenCredentials();
if (!getCredentialsResult)
{
    // The error message set in GetTokenCredentials
    return false;
```

```
}
```

To begin using the REST API we use the PowerBIClient class, this object has the methods to call the Power BI REST API. The following code uses the PowerBIClient class and list the reports in the workspace.

```
// Create a Power BI Client object. It will be used to call Power BI APIs.
using (var client = new PowerBIClient(new Uri(ApiUrl), m_tokenCredentials))
{
    // Get a list of reports.
    var reports = await client.Reports.GetReportsInGroupAsync(WorkspaceId);

   // No reports retrieved for the given workspace.
   if (reports.Value.Count() == 0)
   {
     m_embedConfig.ErrorMessage = "No reports were found in the workspace";
     return false;
   }

   Report report;
   if (string.IsNullOrWhiteSpace(ReportId))
   {
     // Get the first report in the workspace.
     report = reports.Value.FirstOrDefault();
   }
}
```

We can continue using the REST API depending on our needs, for example the following code lists the datasets inside a specific workspace and gets information about roles.

```
var datasets = await client.Datasets.GetDatasetByIdInGroupAsync(WorkspaceId, report.DatasetId);
m_embedConfig.IsEffectiveIdentityRequired = datasets.IsEffectiveIdentityRequired;
m_embedConfig.IsEffectiveIdentityRolesRequired = datasets.IsEffectiveIdentityRolesRequired;
```

The following code uses the Power BI REST API to get the list of dashboards.

```
// Create a Power BI Client object. It will be used to call Power BI APIs.
using (var client = new PowerBIClient(new Uri(ApiUrl), m_tokenCredentials))
{
   // Get a list of dashboards.
   var dashboards = await
   client.Dashboards.GetDashboardsInGroupAsync(WorkspaceId);

   // Get the first report in the workspace.
```

```
        var dashboard = dashboards.Value.FirstOrDefault();

        if (dashboard == null)
        {
          m_embedConfig.ErrorMessage = "Workspace has no dashboards.";
          return false;
        }
    }
```

Depending on your needs you can continue using the REST API using the methods implemented in the PowerBIClient class.

Summary

Power BI Service gives you the user interface to administrative tasks but you can use the REST API in order to integration Power BI with your applications. In this chapter we gave you he steps on how to do configuration to be able to used the Power BI REST API and also, we gave some starting examples on how to use it inside C#.

About the Author

Eduardo Castro is an Enterprise Chief Architect, Microsoft Regional Director and Microsoft Data Platform MVP, living in San José Province, Costa Rica.

With more than 20 years of experience in IT projects in public sector and large companies, Eduardo has helped the companies to achieve their best using technology based on Windows or Linux. A fan of new technology, entrepreneurship, and travel, he enjoys drinking good cup of coffee. Eduardo is a regular speaker in several Microsoft Conferences and Community Events in USA and Latin America.

Chapter 14: Real-Time Streaming Datasets

Author: Manohar Punna

Power BI helps you build appealing visualizations of your data. With an ever-growing footprint of data, it is essential to get real-time insights into your data. These insights can be as simple as monitoring a single metric or as complex as viewing real-time sales performance across multiple locations. Power BI real-time streaming datasets enable you to stream data and update dashboards in real-time. Any time-sensitive data can be a source of streaming datasets such as IoT sensor devices, social media sources, and service usage metrics.

In this chapter, I will introduce different types of streaming datasets and step-by-step implementation of these datasets using various examples. By the end of this chapter, you will be able to create streaming datasets, push data into them, and visualize data from them in Power BI.

Introduction

Real-time analytics has become the norm for most businesses. The application of real-time streaming is useful in monitoring sensor data, social media trends, and metrics from any time-sensitive analysis on data captured in real-time.

With this requirement, there is a need to address the complexity of implementing real-time streaming solutions. Microsoft Power BI provides real-time streaming datasets as a solution to this requirement. Streaming datasets can be created to collect the transmitted data and analyze them using real-time streaming visuals. These visuals can be real-time on dashboards and can also be used in building reports.

The definition of “real-time” varies among industries and businesses. For example, in a manufacturing factory, detecting an anomaly in the product manufactured should be done before the product is packaged and sent. Whereas in the same factory, monitoring pressure levels are critical and need to be fixed within a few minutes. As an architect designing a real-time solution, it is crucial to identify and record these requirements and service level agreements to design a suitable real-time solution.

Real-Time Datasets

In Power BI, there are three types of real-time datasets:

- Push dataset
- Streaming dataset
- PubNub streaming dataset

These datasets are designed to display real-time streaming data on dashboards. First, let us see how these datasets work. Later we can look at pushing data into these datasets and building visuals from these datasets.

Push Datasets

Push dataset, as the name suggests, allows to push data into the Power BI service. When you create this dataset, Power BI provisions a database to host the data pushed into this dataset. This database is only accessible as a dataset in Power BI. There is no way to connect to this database directly.

The data stored in the dataset can be used to build visuals in reports. These visuals can then be pinned to dashboards to display real-time data.

Note: Visuals created using real-time streaming datasets refresh in real-time on dashboards only. The Power BI service triggers the refresh of a dashboard tile when the data refreshes on the real-time dataset.

Once you pin a visual created on a push dataset to a dashboard, you can also perform Q&A in natural language on the dataset. You can pin the resulting visual as a live tile on the dashboard.

It is important to note that when you pin the live page that hosts the visuals created on real-time datasets to a dashboard, the data does not refresh in real-time on the live page.

Advantages and Disadvantages

The main advantage of push datasets is to build visuals in reports similar to standard datasets. The data is saved forever in the provisioned database. Hence, the reports can be built to analyze historical data for in-depth analysis in addition to real-time analysis.

The disadvantage of having to push data into a database is that the data refresh is not instantaneous. You can expect to see a latency of 3-5 seconds for the data to appear on the visual on the dashboard from the time data is pushed from the source.

Data ingestion is at the rate of 1 request per second with up to 16 MB per request. The throughput is limited to 1 million rows of data per hour. If your data is more extensive than these limits, it is recommended to push aggregated data from the source.

Streaming Datasets

A streaming dataset is the pure flavor of a real-time dataset. The latency is minimal and is built only for real-time analysis. The dataset is provisioned in Power BI using a temporary cache. This mechanism helps reduce the latency and provides near real-time data access in Power BI. As this is a temporary cache, the data is hosted up to one hour, hence providing a transient real-time trend for visuals like a line chart.

The visuals can be created only on a dashboard using the *Add Tile* functionality. You cannot create a visual using this dataset in reports. All the report-specific functions like filtering and custom visuals are not available when using the streaming dataset.

Streaming datasets are available under the **custom streaming data** option as a data source and is visible when you create a live tile on the dashboard.

Advantages and Disadvantages

The advantage of using streaming dataset is that it provides very little latency. There is no database to host the data, which reduces the latency of writing and reading from a database.

The disadvantages are that there is no access to historical data, and there are limited visuals that you can create on a dashboard using live tiles. The data is saved temporarily for up to one hour.

Data ingestion is at the rate of 5 requests per second with up to 15 KB per request. There is no throughput limit for streaming dataset, while the size of the request is less than that of push dataset. This limit is set because it targets scenarios where you would want to use streaming datasets to view data which is meaningful for real-time analysis as-is, like temperature readings or other sensor data to detect any spikes.

PubNub Datasets

PubNub is a streaming service provided by PubNub Inc. These datasets are useful if you are already using PubNub for your real-time solutions and would like to integrate it into your reporting in the Power BI service. The Power BI service uses the PubNub SDK to connect to the PubNub data stream. PubNub

hosts the data and pushes to Power BI through the PubNub data stream.

PubNub datasets are streaming datasets where the data does not reside on Power BI. So, you cannot build reports using this dataset. The only available option to use this dataset is to use live tiles in the dashboard.

The refresh of live tiles is almost instantaneous as the service connects directly to the data stream. As the data is not hosted on Power BI service, there are no limits defined on these datasets. The limits are defined in PubNub service.

Creating Real-Time Datasets

Real-time datasets can be created on the Power BI service using the following methods:

- Power BI Service UI
- Power BI REST API
- Azure Stream Analytics
- PubNub

Power BI Service UI

Real-time datasets can be created using the UI on the Power BI Service. You need to follow the below steps to create a new real-time dataset:

1. Click on any workspace in the Power BI portal.
2. Click on **Create** and select **Streaming dataset**.

]

gure 14-01: Create – Streaming dataset

3. Select API to create a **Push** or **Streaming** dataset using UI in Power BI service.

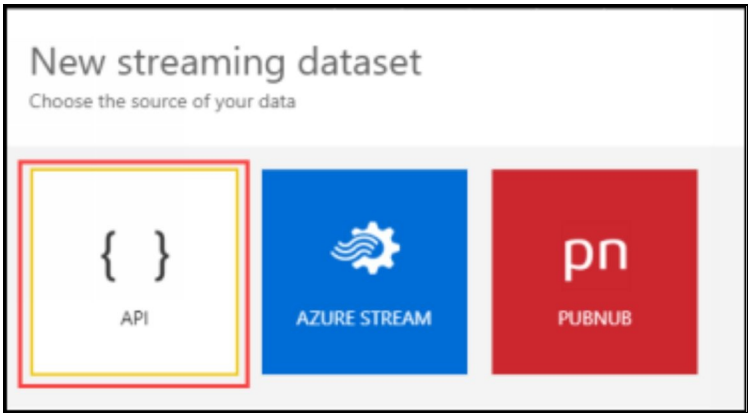


Figure 14-02: New streaming dataset – Choose API

4. In the **New streaming dataset** screen, provide the **Dataset name**. Under **Values from stream**, provide the columns in the dataset with an appropriate data type.

gure 14-03: New streaming dataset

sample JSON string shows up as you type in the columns and the data type. This JSON string is the format for the data that you pass as payload to push data

to the dataset.

ıe option **Historic data analysis** selection decides if the dataset is push or streaming. If this option is selected, the data is saved forever in a database, making it a **push dataset**. If you do not opt to select this option the dataset is saved temporarily in a cache, making it a **streaming dataset**.

ick **Create** button to create the new streaming dataset.

5. On the **Streaming dataset created** screen, the URL to use for pushing the data into the dataset is provided. Also, sample scripts are provided in **cURL** and **PowerShell** syntaxes to push data to this dataset.

gure 14-04: Streaming dataset created

Once the dataset is created, we can start creating visuals in dashboards (and reports in case of push dataset).

Power BI REST API

The Power BI REST API provides the functionality to create datasets and post data to the datasets. The dataset can be created by using PostDataset API using the following URLs for the POST request. The body passed with the POST request contains the structure of the dataset that is created.

My Workspace: *https://api.powerbi.com/v1.0/myorg/datasets*

App Workspace: *https://api.powerbi.com/v1.0/myorg/groups/{groupid}/datasets*

Request Body:

```
{
"name": "Streaming_MVPDD",
"defaultMode": "PushStreaming"
"tables": [
    {
    "name":
"RESTStreaming_MVPDD",
    "columns": [
            {
```

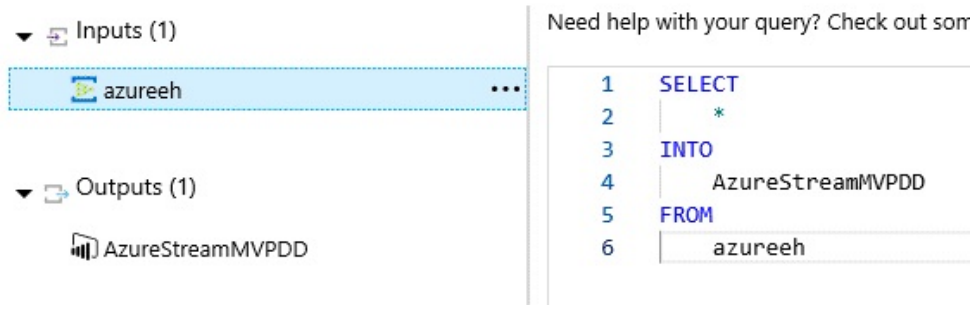
```
            "name": "state",
            "dataType": "string"
           },
      {
            "name": "value",
            "dataType": "Int64"
           }
      ]
    }
 ]
}
```

Azure Stream Analytics

In the first two methods of creating a streaming dataset, you need to create the dataset knowing the structure of the dataset. When pushing data from Azure stream analytics job, the dataset is created based on the data passed from the query output. If a dataset exists in Power BI with the same name, the dataset is recreated, and any structure that exists in Power BI is overwritten with the new structure from stream analytics job.

To create a streaming dataset using Azure Stream Analytics, you need to follow the steps below.

1. Setup an Event Hub to which you can push data using the steps provided here
2. *Create an Event hub – https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-create*
3. *Send and receive events - https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-dotnet-standard-getstarted-send*
4. Setup a Stream Analytics job with Event Hub as the source and Power BI as a target for the query.



gure 14-05: Stream analytics job – Query

5. Start the Stream Analytics job and start pushing the data into the event hub.
6. The Power BI service creates the new dataset when the first set of data arrives into the Power BI service.

gure 14-06: Edit streaming dataset

PubNub

You can create Power BI streaming datasets that connects to a PubNub data stream by selecting **PUBNUB** when creating a new streaming dataset.

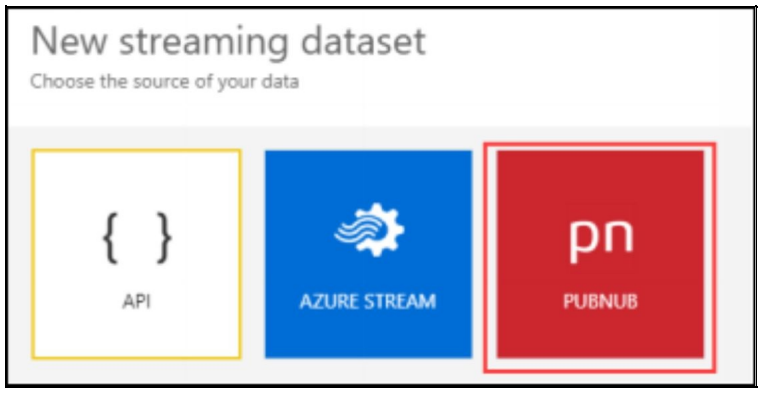


Figure 14-07: New streaming dataset – Choose PUBNUB

To connect to a PubNub data stream, you need to provide the **Sub-key** and **Channel** name from PubNub data stream.

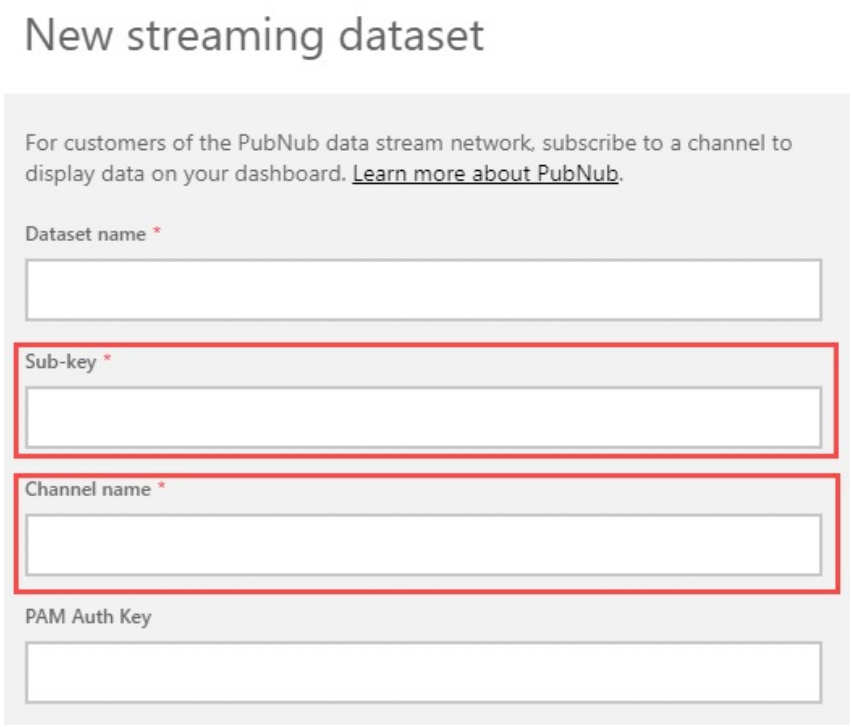


Figure 14-08: New streaming dataset – PubNub dataset

A sample data stream is available to test the PubNub data stream.

Sub-key - *sub-c-5f1b7c8e-fbee-11e3-aa40-02ee2ddab7fe* Channel - *pubnub-sensor-network*

Push Data to Streaming Datasets

Once the push or streaming dataset is created using one of the above methods, data can be pushed using HTTP post requests using Power BI REST API calls. For the scope of this chapter, let us detail the steps to push data using PowerShell scripts with an example.

The sample script provided when creating a dataset using Power BI service UI can be used as a reference to write the post requests. The sample scripts are accessible after creating the streaming dataset by clicking the icon on the dataset.

Figure 14-09: Dataset properties

```
$endpoint = "https://api.powerbi.com/beta/00029a0b-28a4-4b60-aed5-b8b504506e77/datasets/f77(
ec9b2241ab18/rows?
key=Ljvs8eMHX%2F1WvGp4Hq2ejKoCfe4uXqmadPJjIewHdThq0rI1Ch%2B5m7L1AXEj3YdS
$payload = @{
    "state" ="AAAAA555555"
    "value" =98.6
}
Invoke-RestMethod -Method Post -Uri "$endpoint" -Body (ConvertTo-Json @($payload))
```

For simulating a real-time workload, I have a database where I insert the data in a loop. I also have a procedure which reads the data from the table and marks the data as “read” once it has been read. The SQL script for the schema is as follows.

```
--Tables
CREATE TABLE [dbo].[StreamTest](
        [id] [int] IDENTITY(1,1) NOT NULL,
        [EventTime] [datetime] NULL,
        [StateName] [nvarchar](30) NULL,
        [Value] [int] NULL,
        [ReadStatus] [bit] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[States](
        [id] [int] IDENTITY(1,1) NOT NULL,
        [StateName] [nvarchar](3) NULL
) ON [PRIMARY]
```

```
INSERT INTO dbo.States
VALUES
('NSW'),('VIC'),('QLD'),('SA'),('TAS'),('WA'),('ACT'),('NT')


-- Proc to insert data into StreamTest table
CREATE PROCEDURE [dbo].[StreamInsert_prc]
        (@value INT = NULL
        ,@StateName NVARCHAR(3) = NULL)
AS
BEGIN
        --DECLARE @value INT = NULL, @StateName NVARCHAR(3) = NULL
        IF OBJECT_ID('StreamTest') IS NULL
        CREATE TABLE StreamTest
                (id INT IDENTITY
                ,EventTime DATETIME
                ,StateName NVARCHAR(30)
                ,Value INT
                ,ReadStatus BIT DEFAULT 0)

        INSERT INTO StreamTest (EventTime, Value, StateName)
        SELECT --GETDATE()
        CAST((GETDATE() AT TIME ZONE 'UTC') AT TIME ZONE 'AUS Eastern
        Standard Time' AS DATETIME)
        , ISNULL(@value,CAST(RAND()*100 AS INT))
        , ISNULL(@StateName, (SELECT StateName FROM States WHERE id =
        CAST(RAND()*100 AS INT)%8+1))
END
GO


-- Proc to read data for push dataset

CREATE PROCEDURE [dbo].[StreamRead_prc]
AS
BEGIN
         DECLARE @id INT

        SELECT @id = MAX(id)
        FROM StreamTest
        WHERE ReadStatus = 0

        SELECT EventTime, Value, StateName AS State
        FROM StreamTest
        WHERE id <= @id AND ReadStatus = 0
```

```
        ORDER BY EventTime
        FOR JSON AUTO

        UPDATE StreamTest
        SET ReadStatus = 1
        WHERE id <= @id

END
GO
```

Run the following script to insert data in a loop.

```
WHILE (1=1)
BEGIN
        EXEC StreamInsert_prc
        WAITFOR DELAY
'00:00:01'
END
```

To push the data in increments, modify the sample PowerShell script to run in a loop. This script picks the incremental data from the database and pushes it to the streaming dataset in Power BI. This method is the same for both push and streaming datasets.

You can start creating visuals in Power BI, once this data appears on the dataset.

```
#Copy endpoint for push dataset
$endpoint = "https://api.powerbi.com/beta/00029a0b-28a4-4b60-aed5-b8b504506e77/datasets/f77
ec9b2241ab18/rows?
key=Ljvs8eMHX%2F1WvGp4Hq2ejKoCfe4uXqmadPJjIewHdThq0rI1Ch%2B5m7L1AXEj3YdS

#Run in a loop for 100 iterations
For ($i=0; $i -le 100; $i++)
{
   $conn = New-Object System.Data.SqlClient.SQLConnection
   $conn.ConnectionString = "Data Source=<database server>;Initial Catalog=<database>;user=<

   #connect to the database to run the read proc
   $conn.Open()
   $table = new-object "System.Data.DataTable"
   $tableAgg = new-object "System.Data.DataTable"

   $cmd = $conn.CreateCommand()
   $cmd.CommandText = "EXEC StreamRead_prc"
   $result = $cmd.ExecuteReader()

   #Push the json result form proc to streaming dataset
   if ($result.HasRows -eq "true")
```

```
    {
        $table.Load($result)

        $payload = $table[0].Rows[0][0]
        $payload.substring(1,$payload.Length-2)

        Invoke-RestMethod -Method Post -Uri "$endpoint" -Body $payload
    }
    $result.Close()

    start-sleep -s 3
    echo "iteration $i completed"
    $conn.Close()
}
```

The above PowerShell code or code using other supported languages can be integrated into your existing application to push data into streaming datasets in Power BI.

In case of using an Azure Stream Analytics job or using PubNub as the data source, there are no additional steps required to push data into streaming datasets. The Stream Analytics job pushes data to the streaming datasets. The PubNub SDK that connects to the PubNub data stream triggers data refresh on live tiles as the data refreshes in the PubNub data stream.

Visualizing Real-Time Datasets

There are two ways you can visualize streaming data on a Power BI dashboard:

- The first method is using streaming datasets that are created using the UI, Azure Stream Analytics, or PubNub directly on the dashboard to create a tile.
- The second method is to create reports using the push dataset and pin the visuals to the dashboard.

Streaming datasets

1. On a dashboard where you want to visualize streaming data, click on **+ Add Tile**.

gure 14-10: Dashboard – Add tile

2. On **Select source** screen, under **REAL_TIME DATA**, select **Custom Streaming Data** and click **Next**.

gure 14-11: Dashboard – Add tile – Select source

3. On the **Choose a streaming dataset** screen, under **YOUR DATASETS**, select any push or streaming dataset that exists. In this example, let us choose the sample PubNub dataset.

gure 14-12: Dashboard – Add tile – Choose streaming dataset

4. On the **Visualisation Design** screen, select the **Visualization Type**, **Axis**, **Legend,** and **Values** as applicable based on the visualization type.

gure 14-13: Dashboard – Add tile – Visualization design

ıe time window can be adjusted to display data for up to one hour.
ıe types of visuals that can be created on a dashboard using the **add tile** function is limited. You can create the following types of visualizations:

- Card
- Line chart
- Clustered bar chart
- Clustered column chart
- Gauge

5. On the **Tile details** screen, change the **Title** and **Subtitle** as needed. You can also link the visual to an URL or redirect to a report or dashboard within the same workspace.

gure 14-14: Dashboard – Add tile – Tile design

After adding the visual, the data on the visual refreshes in real-time as the PubNub data stream refreshes.

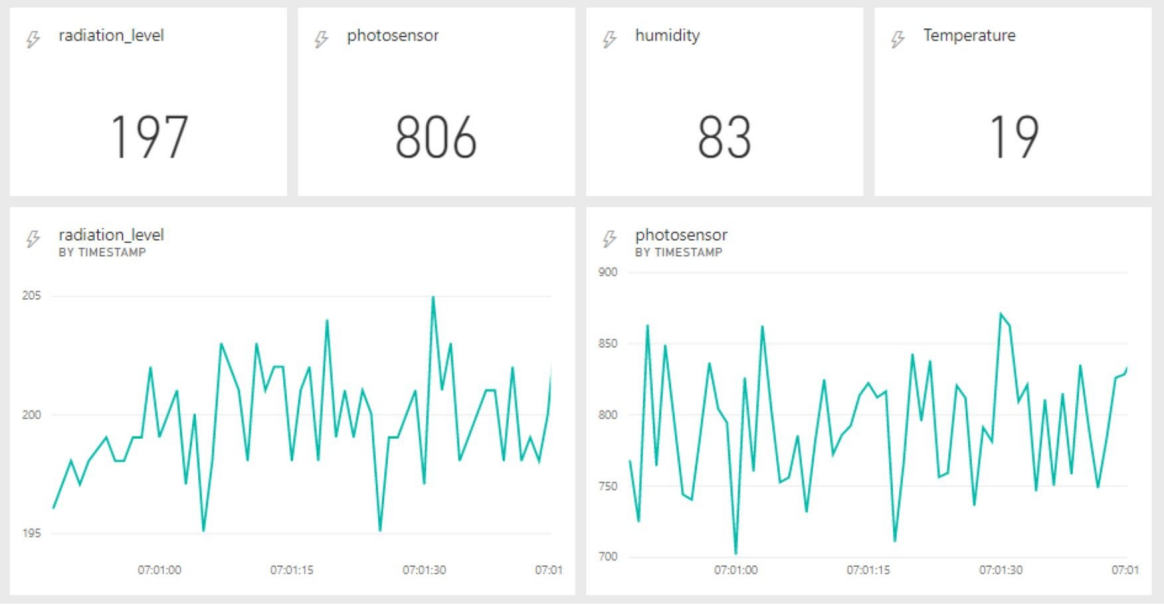


Figure 14-15: Real-Time Dashboard

Push datasets

This method gives you access to more visuals that are not available when creating live tiles on the dashboard. As mentioned earlier, there is a small latency in the data displayed using a push dataset.

1. Click on the **Create report** icon on the push dataset.

gure 14-16: Push dataset – Create report

ote: Only the push dataset has this option, which shows that you can start creating reports using this dataset. Streaming datasets cannot be used to create reports.

2. Create a visual on the report of the type that is not available in the live tile option. For this example, a 100% Stacked column chart is used.
3. Pin the visual to a dashboard.

gure 14-17: Pin visual to dashboard

4. When you push the data into the push dataset using the PowerShell script provided in the earlier section, the data refreshes with a small latency on the dashboard tile.

gure 14-18: Real-time streaming - dashboard vs report

ote – The data does not refresh on the report page. You need to refresh the report page to see the new data.

Summary

In this chapter, you have seen the different types of real-time streaming datasets available in Power BI. You have learned in detail the usage of these datasets, which will help you in choosing the right option for your requirement. You have gained the knowledge of creating datasets and pushing data into these datasets. Finally, you have seen how you can visualize different real-time streaming datasets on a Power BI dashboard.

About the Author

Manohar Punna is a Data Platform Geek and Vice President of DataPlatformGeeks.com by passion and a Data Platform Consultant by profession. He is a speaker at various events around the world like PASS Summit, SQLBits, SSGAS / DPS conferences, SQLSaturdays, SQL Server Day events, and many user groups. He has a wide variety of experience working on code to manage physical hardware, database administration and development, architecting databases for enterprises, designing and building end-to-end BI solutions and building business solutions using PowerApps and Flow. He has authored over 150 blogs and has written One DMV a Day series which is the most extended one-day series on any topic on SQL Server till date.

Part VI: Power BI Usage in Enterprise Environment

Chapter 15: Introduction to Conversation-Centric Design ™

Author: Treb Gatte

Many organizations are challenged when attempting to design business intelligence content for the first time. Where do you begin? How can you ensure your content is adopted? The Conversation-Centric Design approach ensures you design BI content that is aligned to the business need, structured where you can manage scope and ensures that it is clear to the consumer where the content should be used.

“Spreadsheet on a Web Page”

Steve plopped in his office chair and let out a sigh. Dipti, his team manager, seeing him return, walked over to his desk. “Hey Steve, how did the dashboard design meeting go?”, she asked. “It was awful, really.” Steve shook his head and explained. “All the users wanted was a spreadsheet on a web page so that they could see all the details.” “What about the ability to show data graphically in Power BI?” she asked. He replied, “They simply couldn’t see how it applied to them nor could they explain the questions they were looking to answer from the one big spreadsheet.” Dipti frowned. “We will need to try again with this group. If we don’t get them excited by the visualization capabilities, this highly visible project is going to fail and take our jobs with it.”

What’s really going on here?

This scene happens over and over in many workplaces as they start their Power BI journey. We approach Business Intelligence (BI) content development as a blank canvas exercise. Unfortunately, most people are not Picasso and struggle to visualize a wonderful solution to their BI needs from a blank canvas. They need help to understand what decisions are important to make and how does that map to their BI content so that they know where to use their brand-new reports and dashboards.

Figure 15-01: Artist, Jackson Pollack at work

In the end, many design their BI content the way the artist Jackson Pollack created art. Pollack would throw paint at the canvas and a masterpiece would emerge. Unfortunately, using the same approach with data ensures that you deliver many reports and dashboards, but few answers.

Conversation-Centric Design™ Overview

The Conversation-Centric Design™ (CCD) process uses the user's social interactions to provide context and direction for the development of Business Intelligence content.

All work is inherently social within the workplace. We interact socially when:

- We create work
- We are assigned work
- We have questions about the work
- We report status on the work
- We deliver the outcomes.

These interactions are called client calls, meetings, projects, hallway conversations, etc. Many of these interactions are formal in nature. They occur regularly, have a set attendee list and a set agenda. Hence, the CCD process leverages formal interactions as a starting point for BI content design.

Why formal interactions?

Successful adoption of new BI content is key to your project's success. Projects that change the way a person works fail when it is unclear to a person where and how to use the new content in their work. Using these formal interactions makes it clear to the impacted person where to apply the new functionality.

Process Overview

The Conversation-Centric Design™ process has four distinct phases as shown in the figure below. The intent is to take you from a position of endless possibilities to a starting point with clear priorities and steps to deliver them.

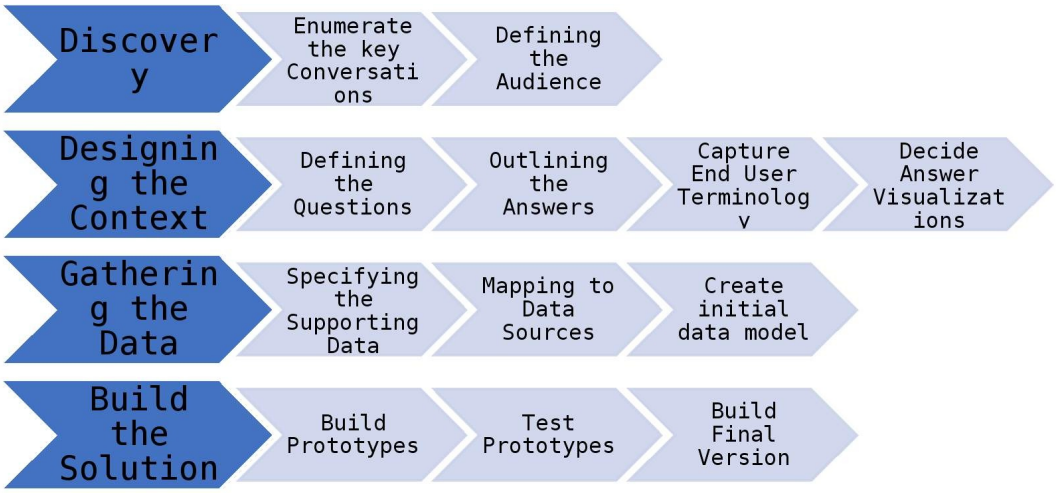


Figure 15-02: Detailed Conversation Centric Design process

Discovery

The discovery phase provides the steps necessary to focus your efforts on the key conversations. First, we start by cataloguing the key conversations. This can be standing meetings or common interactions like status reports or client meetings. The goals are to create a recognizable list of conversations and a list of audience members for each conversation.

The conversation list is used for four purposes.

Scoping

This use enables you to decide which conversations to invest in from a business intelligence perspective. One of the hardest decisions for business decision maker (BDM) to make is around scoping as many data investments are opaque to them. Providing a known event as the scoping mechanism means the BDM understands what the data investment impacts and the relative importance of the event. This context enables the BDM to make an informed scoping decision.

Prioritization

This use enables the BDM to decide which conversations to address first. This aspect enables the BDM to get the biggest impact for the investment. If they are hoping to win support in the executive suite, prioritizing the executive conversations first ensures the audience's needs are met.

Status

The business event list provides the context for the current activity and status. BDMs are usually not data professionals so providing status on data models, measures, and Extract-Transform-Load (ETL) processes without context, results in glassy eyed looks, nods of the head, and zero understanding of what's really happening. By providing updates relative to the business event, it helps the BDM in understanding where the impact of a given topic will lie. We should still take steps to make the process as clear as possible for the BDM.

Audience

Capturing the audience per conversation forms the basis for a security model and for testing groups later in the process. It also provides a list of stakeholders for reporting status.

Many business intelligence projects only consider the direct requirements of the requestor. However, once implemented more broadly, a flurry of activity results as other stakeholder needs are discovered and were not accommodated in the design. This can lead to expensive rework.

Using the STAR model below, you can consider or purposely eliminate the needs of various potential stakeholders. Doing so ensures you are correctly considering a broader audience if needed.

Figure 15-03: STAR model for potential audience discovery

Example of Discovery outcome

Conversation	Frequency	Audience	Priority
VP meeting	Monthly, usually first Wednesday of Month	Joe, Pradeep, Karthik, Amy, Mason	1
Project Status Meeting	Weekly on Monday	PM and PM Team	4
Finance Review	Monthly on first Tuesday of Month	Bill, Susan - Accounting, Joe, Pradeep, Karthik, Amy, Mason	2
3 + 9 Finance Exercise	Quarterly	Bill, Susan - Accounting, Joe, Pradeep, Karthik, Amy, Mason	3

Designing the context

Once we have our prioritized list of conversations from Discovery, we start to break down the activity into our four areas. These areas are:

- Defining the questions
- Outlining the answers
- Capture end user terminology
- Decide answer visualizations

Developing Key Questions

Conversation-Centric Design starts with a key conversation. We'll discuss how to decide what those will be shortly. For example, let's use the Monthly VP Meeting as an example.

Using a meeting like this, we gain a lot of useful information quickly. First, we know what the conversation generally is about as there is an agenda for this meeting. We know the audience that attends the meeting. Looking at the figure below, we can see that two items are already captured.

When you drill into the meeting agenda, rephrase the items as a series of key

questions. For example, your list could look like this:

Original agenda item	Question version
Budget Update	Where are we with regards to our overall budget? What are our top five deviations from budget?
Key Projects Update	What projects are running late? What projects are finishing this month? What projects are starting this month?
Personnel Update	What roles are short-staffed over the next six months? What roles are being filled primarily via vendors?

The next step is to group, prioritize, and triage the questions as it may not be possible to address all questions due to resource constraints or other factors. Optimally, this should be done by the attendees. They should rank the questions in importance and the input should be used to stack rank the questions.

Developing the Answers

Developing the answers for each question requires deciding several aspects to ensure you have what you need. The goal is to uncover mismatches between requirements and process.

For example, one VP has determined that the Budget reports need to be refreshed daily. However, the process that delivers the budget data only runs monthly. Therefore, do you as the creator want to face a VP who has been looking at the reports daily expecting changes or do you want to make them aware of this issue upfront?

Data

In examining the questions, a key need is identifying the source and data elements needed to address the question. For example, the question “Where are we with regards to our overall budget?” implies that the budget numbers are in a system that can be accessed and can be matched with expenditures in some fashion.

Definition

When formulating the answers, it is very important to clearly define the conditions that influence or are used in filters, KPIs, measures, conditional columns, and other constructs. As you will see later in the chapter, this may be the most important aspect in ensuring expectations are met and all data consumers have the same shared understanding of the data represented.

Granularity

Once we determine that the data exists and can be accessed, the next question is one of granularity. How granular does the answer need to ultimately be? This will create the basis for groupings, pivots, and what's the lowest level of data available to view.

Process

We also need to understand the process by which the data is created and delivered. In some cases, direct access to the data source enables faster access. However, high business impact data such as financial information may only be provided on a recurring basis, in a summary format like a CSV file. The timing and process by which the data is delivered can significantly impact delivered functionality. It also creates a data requirement to include the last modified date of the data to the data consumer so that they are aware of the data's age.

Terminology

Power BI has a powerful ad hoc querying tool called Q&A. Q&A depends on a detailed linguistic schema to map data consumer terminology to data elements in the Power BI Model so that it knows what to data to query when asked a question. As you are developing and designing questions and answers, it is important to capture the language used by the data consumer when referring to a data element. This can be incorporated into the model's linguistic schema, making Q&A much more useful. For example, if I have a column called ProjectName in the data source but my consumers call them Gantts, schedules, and plans, I'd incorporate those three terms in the linguistic schema so that Q&A would understand the consumer request.

Tool

Microsoft provides three core tools for Business Intelligence: Power BI, Power BI Paginated Reports, and Excel. It is imperative to pick the right tool for the need to prevent brittle, workaround solutions. For example, if there's a need to print a report to provide to field personnel, it may be a better fit for Power BI Paginated Reports rather than attempting to engineer a solution in Power BI

itself.

Visualization

Once you have the question and the answer with permutations decided, you can classify your answer to determine the best set of visualizations for the need. By permutations, the answer may require different groups, levels of rollup, etc.

There are seven types of visualizations.

- Nominal comparison
  - Comparison of quantitative values grouped by some categorization value.
  - Example: Headcount by Department
- Time-Series
  - Changes in one or more metrics over a defined time period
  - Example: Year to date expenditures
- Ranking
  - Comparison of two or more values to illustrate relative contribution in a most to least or least to most pattern
  - Example: Top 5 projects by total budget
- Part to Whole
  - Comparison of a data subset to the whole data set
  - Example: Percentage of riders of a specific bus line compared to overall riders
- Deviation
  - Comparison of data points against expected data values
  - Example: Monthly actual sales versus projected sales
- Distribution
  - Visualization of data distribution, either around a mean, timeframe or some other nominal value
  - Example: Product demand by age group
- Correlation
  - Changes between two or more metrics that change in relation to each other in a positive or negative way
  - Example: Job performance ratings versus social likeability scores

Visualization types are as follows:

Type of Visual	Example	Use Lines to	Use Bars to	Use Points to	Suggested Power BI Visual

Nominal Comparison	Number of hours logged this week toward a project versus planned hours for the week	Show deltas between values and overall trend	Compare between individual values	To show individual values. Add a line to highlight the trend	Column Chart Line Chart Table Matrix
Time-Series	Project Costs by Month	Show trends in the data	Compare between individual values	To show individual values. Add a line to highlight the trend	Column Chart Line Chart
Ranking	Top 5 Projects by ROI		Show ranking, which is the most common use of bars	To show ranking using a nonzero based scale	Funnel Bar
Part to Whole	Annual Project Investment by Strategic Objective		In place of Pie Charts to improve usability		Waterfall Tree Map Stacked Bar Donut Area
Deviation		Show overall trend	Highlight magnitude of deviation	Denote values and use with a line to show trend	Column Chart Gauge Chart Scatter Plot
Distribution		Visualize if the data shape is most important to communicate	Visualize if the individual values are the most important to communicate		Column Chart Area Chart
Correlation		Use with scatter plot to denote trend		Use to denote values	Scatter Plot Bubble Plot

Gathering the data

This phase involves the creation of the initial data model in Power BI or could be used to modify an existing data model. The goal is to ensure

- The list of data sources is known so that security access can be requested

- The requisite data is available from the identified data sources
- Create the initial data model

Specifying the supporting data

A review of the answers outlined above provide an opportunity to determine the source of the answer's data.

If your group owns the data, then follow your normal process to connect to the data.

If the data is coming from an external system source that is managed by another entity, access to the data source should be requested. This should also include the creation of data access accounts with requisite licensing. It is a good idea to be engaged with the data source's change management process so that as they upgrade and maintain their systems, your process is not broken by inadvertent changes.

If the data is being sourced from a manual creation process, there is a need to standardize the location and name of the requisite data files. A change management process for the data structures should also be considered for ongoing management. Doing so ensures that inadvertent changes don't break the data refresh going forward.

Mapping to data sources

Once all data sources have been identified, three aspects must be rationalized to ensure use of the data will lead to actionable data.

In many cases, you will need to combine data from the various sources to create a model. Doing so requires a set of common key values between tables. For example, if you are tracking a rolling total of expenditures and need to compare them to the budget, you'll need a key such as budget code on both the budget and expenditure records to tie them together.

In some cases, a multi-part key, involving multiple column values, must be used to denote a unique record. To ensure Power BI can work with this effectively, you'll need to synthesize a compound key, where all the values of the multiple columns are concatenated into a single column value.

Differences in data granularity is another difficulty encountered. Pulling data from various sources are usually at different levels and types of granularity. This creates issues rationalizing the data. For example, your budget information is only available by cost center and your expenditures are at the transaction level. How would you tie these together to create a project level summary?

Mismatches in data refresh cycles can also lead to integration issues. Some data may be real-time, others refreshed once a month. As you bring the data together, a time scale minimum granularity should be defined.

Create initial data model

A data model is ideally structured in a star schema pattern. By this, a model generally has facts and dimensions. Facts are the events that have occurred. Examples of this can be transactions, timesheet entries, status updates, etc. Dimensions are those entities to which the fact applies. For example, for a transaction, dimensions could include billing company, cost center, project, etc.

Once the core data is in place and structured correctly, the initial measures, summary tables and other transformations should be put in place. In many cases, a company may have standard elements that are created for a dimension and/or a fact table. These could include a date table, for example. This creates the basis for the next phase, where iterative development is used to finalize the model and requisite reports.

Build the solution

The building of the solution starts once you have a solid design. This should reduce the time to develop the solution and reduce any rework. Note, the following describes an iterative approach to developing business intelligence content. Many data consumers do not have the training or experience to formulate the final visualizations. The iterative process gives them a voice in the process as well as builds champions for the final implemented product.

Note, the process below is designed for large business intelligence projects with high visibility in the organization. The process can be streamlined for smaller projects as needed. The key learning is to drive the test process rather than simply reacting to it. Simply reacting leads to scope creep, unmet expectations and generally longer timelines to deliver.

Build Prototypes

The initial report prototypes validate the design process findings and are the first opportunity for the data consumers to be hands on. A focus group of data consumers should be identified from the earlier audience identification and recruited to help with this validation.

In most cases, the data author should be listening only, with the session facilitated by another with no vested interest in the feedback. Any feedback that is gathered, should be reviewed, triaged as to whether it is of value to address,

and then work assigned to incorporate feedback.

To prevent runaway scope, there should be a set number of review rounds initially and a data consumer representative should be part of the feedback triage process.

Test Prototypes

Once you've reached a specific level of quality with the model and reports, an extended pilot group should be given access to it to test the new content.

It is imperative that you provide clear guidelines to the new pilot group on how to file feedback, what is considered a bug, what is considered a change, and what will be done if it's simply product behaviour. There should also be a task plan with clearly defined assignments. Otherwise, you'll have a new group of people wandering around in the new content with no context and focusing on product behaviour. This will yield feedback of little value.

Typically, a few rounds are necessary to get everyone's feedback and to have the feedback triaged and prioritized.

Build Final Version

Once the test phase is over, the final build out of the content occurs. The work plan of remaining items should be tracked. The content should be packaged as an app and released according to internal procedures to the target audience defined earlier.

A Real World Example

Steve, a business analyst in the IT group, received the following request from Donna, an IT Director.

"Steve, we need a dashboard for our IT Managers and Finance personnel to use in the Monthly Review meeting. This will be used to see project related data. How soon do you think you can get that done?"

Steve sighed and thought, "It's time to put this CCD training to work."

First step is to enumerate the conversation. He highlighted the phrase as follows, as this gave him the conversation and the timing.

Steve, we need a dashboard for our IT Managers and Finance personnel to use in the ***Monthly Review meeting****. This will be used to see project related data. How soon do you think you can get that done?"*

Next step was to determine the audiences. Steve highlighted the following.

Steve, we need a dashboard for our ***IT Managers and Finance*** *personnel to use in the Monthly Review meeting. This will be used to see project related data. How soon do you think you can get that done?"*

Steve frowned, "What I don't see is any actual questions in this request." He replied to Donna with some questions about what questions needed to be answered for the IT Managers and for Finance. This dialog continued for a bit. As he learned in his training, he needed short questions that started with Who, What, When, Where, Which or How much. At the end of the exchange, he had the following questions.

Audience	Monthly Review Conversation
IT Management	Which projects are behind? Which projects are finishing this month? What are the upcoming key milestones this month?
Finance for IT	What projects are over budget? What are the anticipated capital expenses for this month?

Steve decided to focus on the first question to train Donna in what he needed, as this was her first time asking for content. Once she understood what he needed,

the process would go quickly.

Since this was an extension to an existing model, he knew the data would be coming from Project Online. However, he still needed to identify the data fields once this request was clearly defined. He looked at Project and saw there was many projects. Also, what did they mean by behind?

He replied with the following questions.

With regards to the What projects are behind question,

- *What kind of projects should be included in the report? Capital? All?*
- *Define the criteria for "behind. "*
    - *Is it Planned > Baseline?*
    - *Is it some percentage variance?*
    - *Is it schedule or cost?*

After several iterations, they finally arrived at the following criteria.

- Show all capital projects over $250,000 in total spend where the planned finish date is greater than 30 days past their baseline finish date.

Steve gave some thought as to how this might be visualized. He decided that three levels of data were appropriate. This would provide several pivots without being overwhelming. This would also set up the report to enable drill through to other detailed reports.

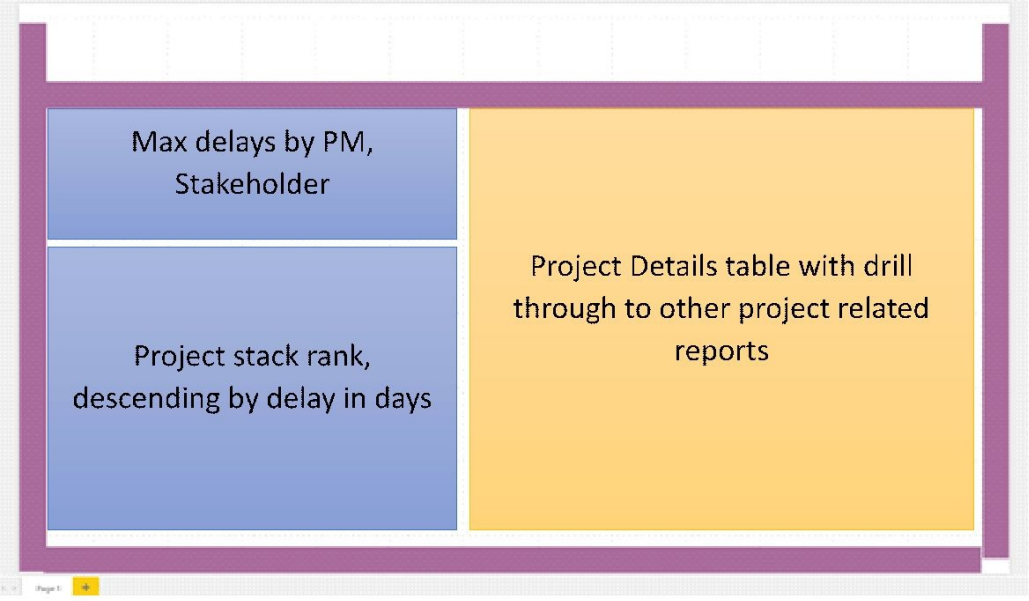


Figure 15-04: Page layout for new project report

He decides to use cards for the max delay information since it's an overall number. He'll use the Stacked bar chart visual to do the project stack rank by delay visual. Finally, he's using the table visual to show key information for each individual project.

His first prototype looks like the following. The entire page is structured around a single question and the number of visuals is kept to a minimum. He repeats this process for each of the questions.

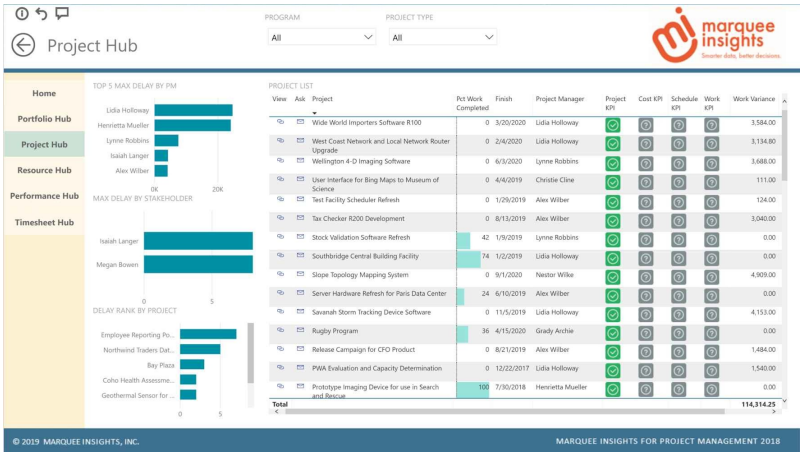


Figure 15-05: Screenshot of prototype

Steve reviews the prototype with Donna. She sees how the reports are question focused and understands how these can be easily used and explained. She asks for specific drill throughs to other reports which Steve can do. She also appreciates the ability to add new pages as new questions arise, without having to re-architect every report.

Once Steve is done with the changes, Donna signs off on it. He publishes the final version to a new workspace in PowerBI.com. This is used to create the **IT Monthly Meeting** app, which Steve publishes and assigns security rights to the IT Managers and Finance.

Summary

The Conversation-Centric Design™ process streamlines the development of business intelligence data by providing a framework that helps you anticipate and address issues before they arise in the development process. Once a solid base of content is in place, it also makes it easier to maintain that content over time as the requisite documentation is created as the content is developed.

Real world use also shows it increases adoption of content since the delivered apps are aligned with the business activity that drove the need for the content. It also reduces the risk of changing reports since the exposure of a given report collection is limited to a single business activity.

For questions about Conversation-Centric Design™, contact us at hello@marqueeinsights.com.

About the Author

Trebuel Gatte, CEO, MarqueeInsights.com

Treb Gatte is a business intelligence expert with 24 years of experience.

Prior to becoming CEO, he worked in leadership positions at Microsoft, Starbucks, Wachovia (now Wells Fargo). He has been recognized as Data Platform MVP from Microsoft.

He lives in the Seattle area with his wife, children and corgi and holds an MBA from Wake Forest University.

Chapter 16: Understanding when to move to Power BI Premium

Author: Gilbert Quevauvilliers

When looking at Power BI Premium there are a lot of options available to you with Power BI Premium. At times when there are too many options and it can potentially be a challenge to understand which features are applicable for your situation. In this chapter you will gain a better understanding of what these options are. By having a deeper understanding of the Power BI Premium features, it will allow you to make a more informed decision on looking to move to Power BI Premium.

Whilst the list below is not comprehensive it does overview the standard set of features that are currently available in Power BI Premium as at July 2019 (Please refer to https://docs.microsoft.com/en-us/power-bi/service-premium-what-is)

Dedicated Performance

When looking to move to Power BI Premium, one of the advantages is having dedicated performance. What currently happens when you are using the Power BI Service you are using a shared environment.

If I had to put this into an example, it would be you are sitting in an aeroplane as you to today. There are a lot of people in the aeroplane, but you are sharing not only the costs of having to fly the aeroplane, but you too are sharing all the components (engines, fuel, entertainment, pilots, air hosts, etc.) that make up the aeroplane.

Whilst when you move to Power BI Premium you are now flying on your private jet. Here you do not share anything with anyone else; the entire aeroplane is available for you to use as you so wish.

The same applies to Power BI Premium, where you now have dedicated CPU and Memory.

At the time of writing this book (July 2019), there are the following dedicated capacities for Power BI Premium, which indicates how much memory, CPU, and Storage per capacity that you purchase.

The details below have been taken from the Power BI Whitepaper Deploying and Managing Power BI Premium Capacities (https://docs.microsoft.com/en-us/power-bi/whitepaper-powerbi-premium-deployment)

Capacity Nodes	Total v-cores	Backend v-cores	RAM (GB)	Frontend v-cores	DQ/LC (per sec)	Model Refresh Parallelism
EM1/A1	1	0.5	2.5	0.5	3.75	1
EM2/A2	2	1	5	1	7.5	2
EM3/A3	4	2	10	2	15	3
P1/A4	8	4	25	4	30	6
P2/A5	16	8	50	8	60	12
P3/A6	32	16	100	16	120	24

Figure 16-01: Table showing different Power BI Capacities

A quick overview of what each of the Capacities is:

- EM – This is for Embedding Power BI into an application (An

application is your custom application that is not part of the Power BI Service).

- A – This is also for embedding Power BI into an application (An application is your custom application that is not part of the Power BI Service). The difference here is that it is run from Azure. The advantage of running it from Azure is that you get the Azure functionally to start and pause the embedded capacity as is required. You can also scale up and scale down the capacity as required.
- P – This is the Premium capacity where it can be used within the Power BI Service and assigned to specific app workspaces, my workspace (users' workspaces) or for the entire organization. One additional thing to note is that the P Capacity can also be used for embedding into an application.

The current way to purchase the Power BI EM or P Capacities is through the Office 365 portal. This is because Power BI currently falls under the Office 365 stable of products.

If you are looking to use the A Capacity that can be created through the Azure Portal.

Here are some of the reasons you might be looking to move to Power BI Premium for the dedicated performance.

Free Users

What happens is that very often, there is a dataset or a report that can assist a large number of users in your organization. You do not want to license each user individually to simply view and interact with a report.

By using Power BI Premium, you can use the feature of free users who can view the content in an App Workspace that has been assigned to Power BI Premium (This only applies on the P SKU)

What could also be considered is if there is a pricing breakpoint between the number of users who need to view dashboards and reports when compared to purchasing Power BI Pro licenses. It is a good exercise to investigate as there might be some great cost savings.

XMLA End Points

XMLA endpoints is fancy terminology for meaning that you can connect to your Power BI Premium dataset with other tools such as Excel.

The way I would explain XMLA endpoints from the standpoint of reporting is

that any client tool that can currently connect to SQL Server Analysis Services (SSAS) Multidimensional will also be able to connect to Power BI Premium capacities.

As it stands today July 2019, the XMLA endpoints are ready-only. In the future are plans for the XMLA endpoints will change to read-write, which will allow for a lot greater management of your Power BI Premium capacities.

If you are looking to leverage other tools which have connectivity to SSAS cubes such as Excel, Tableau, or others, then this is an option to consider.

Higher Refreshes

Currently, when using Power BI Pro, you are limited to 8 refreshes a day. When you have an App Workspace assigned to Power BI Premium you can then have up to 48 scheduled refreshes per day.

This means that during the core business hours (being 8 hours a day), you could refresh the dataset every 15 minutes.

One thing to note is that you would need to ensure that your dataset will have completed refreshing within the 15 minutes time frame, for the next refresh to start.

With higher refreshes, you can refresh the data as often as you like when using the Refresh API.

There are some other considerations to think about when refreshing data, which will be covered next when using Incremental Refreshing.

If you have data that is frequently updating this enables you to do this with Power BI Premium.

Incremental Refresh

Before I get into incremental refreshing, I think it will be good to over off what incremental refreshing is.

The best way to describe it is by using another example.

When you import sales data into Power BI, it all gets imported into one big table as shown below.

Figure 16-02: Complete Sales Data table

If we had to take a closer look at your sales data, it will typically always have a date for when the sales took place.

If we had to look at the same sales data table in another view, it looks like this too.

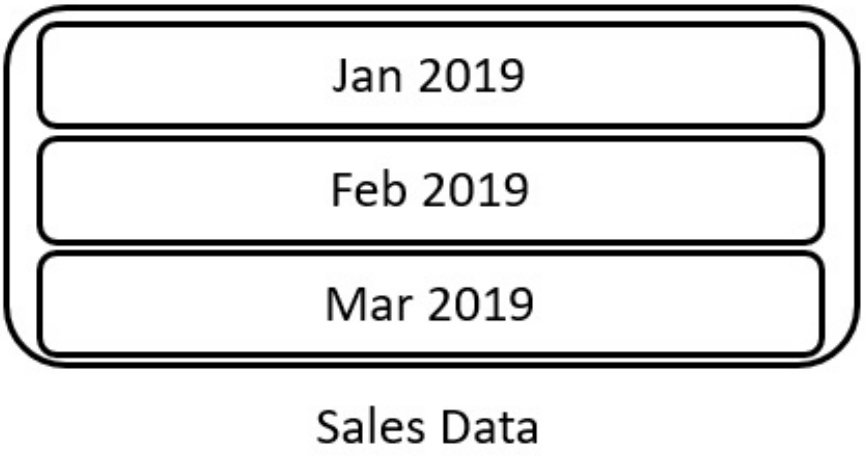


Figure 16-03: Sales Data table with months (partitions)

What you see above is that even though the sales data is still in one table, there are three segments of data. One for each month where there is data, being Jan – Mar 2019.

What incremental refreshing is, it will NOT refresh all the data, but rather

refresh the latest data? Which means that it will incrementally refresh the new or updated data. And not refresh the entire sales data table.

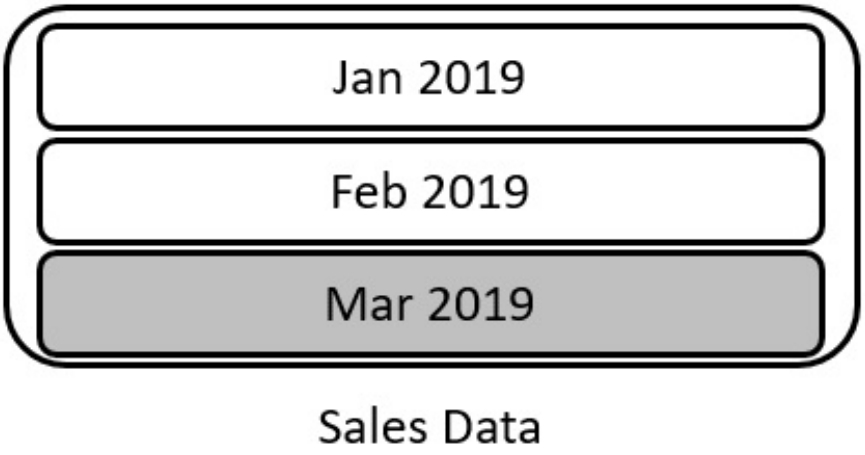


Figure 16-04: Sales Data table highlighting the month of Mar 2019

Using the above example if our current month we were in was Mar 2019, when configuring the incremental refreshing, it would only refresh the Mar 2019 portion of the sales data table.

From the above image, each of the months is known as partitions when using incremental refresh.

There are some distinct advantages of using incremental refresh

Refreshes are quicker

Because you now do not have to refresh the entire dataset, but only a subset of the data it will refresh a lot faster.

This is because if you had a table with three years' worth of data, it would have to refresh 360 000 rows (The number is derived by assuming that each month has got 10 000 rows).

Whilst if you only had to refresh the current months' worth of data, that would mean only refreshing 10 000 rows (if it was a complete month, or even fewer rows if it was during the month).

If we had to put this into a comparative chart, you could see how much less data there is to refresh.

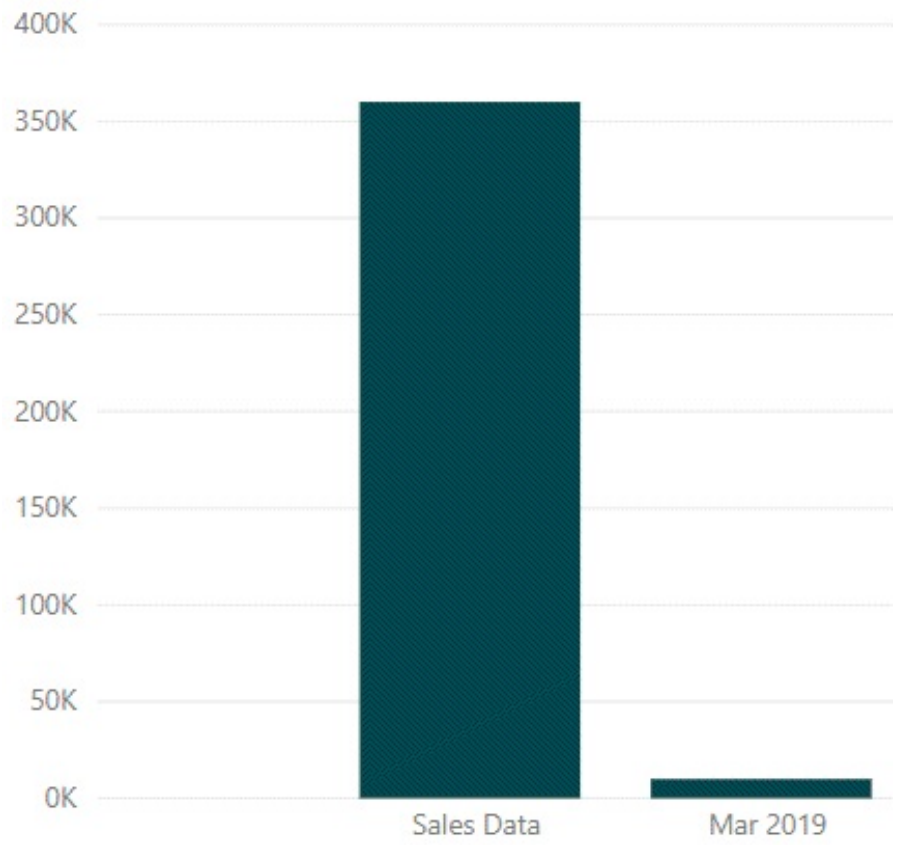


Figure 16-05: Comparison for refreshing the entire Sales data table vs. refreshing the Mar 2019 partition

Fewer resources are required

One of the things to always be aware of, as is typically the case when using dedicated resources is that you only have a certain amount of resources that you can utilize.

When using incremental refreshing, it will use a lot fewer resources if you had to compare it to refreshing the entire sales dataset.

If we have a look at the chart above and assume that it represents resources, you can very quickly see that March 2019 will use a lot less resources.

One thing to note is that as with all things in Power BI, it will depend on a lot of factors in how much resourcing saving you will get, but without a doubt, you will use a lot less resources when using incremental refreshing.

The image below you can see how the incremental refresh uses a lot less memory compared to a full dataset refresh

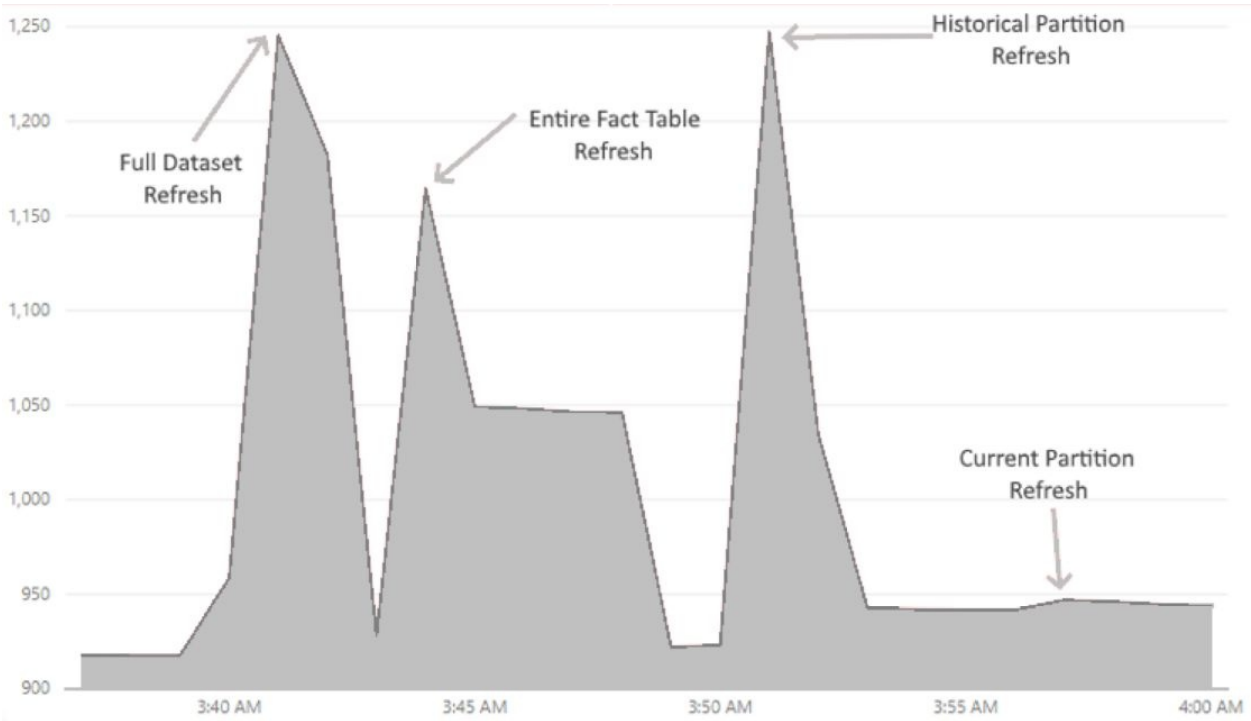


Figure 16-06: Chart showing how different dataset refreshes affect memory consumption

If you are using larger datasets and looking to get the data refreshed as quickly as possible incremental refreshing could be a solution.

Another reason to decide to use Incremental refreshing is if you want the refresh to consume fewer resources, mainly memory and CPU, this could be a viable option to allow that to happen in a very efficient manner.

Dataset size larger than 1GB

When looking to use Power BI Premium another reason to use it, is that it allows you to use datasets that are larger than the 1GB file size limit when using Power BI.

Whilst the Vertipaq engine (this is the compression engine that runs Power BI) is very good at compressing data a lot of organizations have a lot of data, and it simply will not fit into the 1GB PBIX file size, which is a limitation in the Power BI Service not on the actual PBIX. This is where Power BI Premium could be a solution for you.

As shown previously and shown again below, there are different capacities that you can buy based on how large your dataset is.

Note, the image below only shows the Power BI Premium Capacities for the P capacities. This is because these are the only ones that can be used within the Power BI Service.

Capacity Nodes	Total v-cores	Backend v-cores	RAM (GB)	Frontend v-cores	DQ/LC (per sec)	Model Refresh Parallelism
P1	8	4	25	4	30	6
P2	16	8	50	8	60	12
P3	32	16	100	16	120	24

Figure 16-07: Power BI Premium P Capacities

Another thing which you must take note of is when moving to Power BI Premium it is not the size of the PBIX that determines how many resources you consume, but rather how much memory the PBIX consumes once it is loaded and people are using the reports or dashboards used in queries when interacting with the data.

To determine how much memory your PBIX is going to consume, there are two ways to determine this.

As a working example, the size of my PBIX is shown below

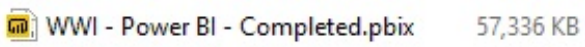


Figure 16-08: File size of the PBIX when stored on disk

Actual memory consumption

A better way to see how much memory the PBIX will consume is to find the

msmdsrv.exe

You can complete the following steps below.

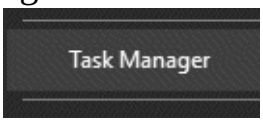
- Make sure you only have got **one Power BI Desktop File Open.**
- Now right-click on the taskbar and select Task Manager
  - 

Figure 16-09: Task Manager selection

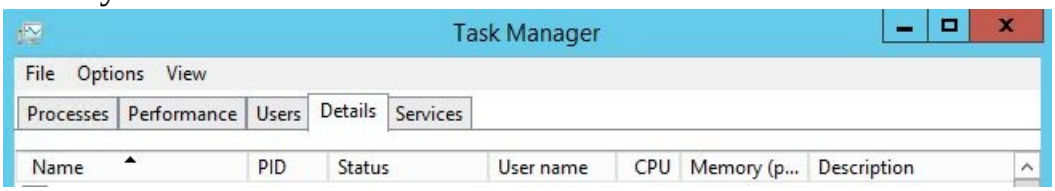
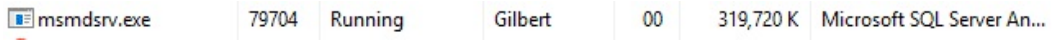
- Next click on the Details tab.
- The easiest way to find the msmdsrv.exe is I click on the Name to sort it alphabetically
  - 

Figure 16-10: Task Manager Window sorting by Name

  - I then click on any item under Name
  - I then type “ms” which allows me to then go down to a name that starts with “ms”
  - I can then **find** my **msmdsrv.exe**
- As you can see below my current PBIX is using about 318MB of memory
  - 

  - **Figure 16-11: msmdsrv.exe memory usage when using Task Manager Window**
- NOTE: Even though it is currently using 318MB of memory I am currently not interacting with the report, which would very likely increase the amount of memory consumed. Not by a significant amount, but just something to be aware of.

A quick tip is if you want to understand what is consuming all the memory in your PBIX, you can use the Vertipaq Analyzer.

This is a tool that has been created by the great people from SQLBI.COM

Here is the link to where you can find their blog post:
https://www.sqlbi.com/articles/data-model-size-with-vertipaq-analyzer/

Part of this blog post also has a link to the Excel file that you can download and allow you to see which columns are consuming a lot of memory. Another use

case is you could use the XMLA endpoints to see how much space is being used.

There are times when either certain columns can be removed (because they are not needed) or they can be changed, which can drastically affect the amount of memory consumed.

The Microsoft Power BI team have indicated that they will be removing the limit on dataset sizes which is currently set as at July 2019 to 1GB for Power BI Pro Users and 10GB for Power BI Premium.

There is also an upcoming feature where they are going to allow you to potentially store dataset sizes of up to 4TB. More on those details will be available on the Power BI Roadmap.

When looking into larger dataset sizes as a reason to move to Power BI Premium it is often a use case that not only do you need the larger dataset size, but it almost goes hand in hand with incremental refreshing.

Paginated Reports

And currently, a lot of organizations have invested a significant amount of time in the creation and development of SQL Server Reporting Services (SSRS).

These were originally the first type of reports that allowed the business users to run their reports, change the parameters, and self-serv.

The SSRS reports were also very good at creating pixel-perfect reports, which allowed for the printing of the reports for a lot of business users.

Not only that, but the business users could also export the data in the reports into multiple formats.

And finally, the SSRS reports could be set up with a schedule to allow the SSRS reports to be emailed to you. This meant that the business users could get the SSRS reports either in the Inbox or from a folder location.

While this is great for On-Premise solutions, there is also the requirement to have all the above functionality within the Power BI Service.

It was called Paginated Reports when moving to the Power BI Service; this was first announced on 11 July 2018 at the Business Applications Summit.

Paginated then went Generally available on 09 June 2019.

As with all things being migrated from an On-Premise solution to the cloud, they do take time to get there. And as has been the approach with Power BI is that they release new products or features with limited functionality and then build new features as defined by the customers, idea’s and their roadmap.

As it stands today, paginated reports do not have full feature parity with SSRS. The paginated reports team has confirmed that their goal is to have all the features in SSRS available in paginated reports.

When looking to move SSRS reports to paginated reports, you would have to first consider if paginated reports have the features that you are looking for. After which you would then need to plan to move your SSRS reports to paginated reports.

One of the major advantages of having paginated reports in the Power BI Service is that you now have one central location for all the reporting requirements in your organization. You could also look at using Power BI Report Server which is the On-Premise Version of Power BI and SQL Server Reporting Services (Paginated reports)

This simplifies where and how users can get the data that they need.

Paginated reports also use existing features and shared and certified datasets, which can be consumed by paginated reports. By having a shared dataset, it means that both Power BI reports and paginated reports will be able to use a single data source for multiple reports. This goes a long way to ensure that when multiple people are looking at data or making decisions, no matter what format they are using the numbers remain the same.

Geographic Distribution

Another interesting feature that is only available to Power BI Premium is the geographic distribution of your data.

What geographic distribution means is that you can physically host your data in a Power BI Premium App workspace in another geographic location.

Typically, there are two reasons why you would be looking to using the geographic distribution feature.

Data Sovereignty

Data sovereignty is another word for ensuring that your data is hosted in the country or location which has been defined by a company or country regulations.

This often happens when certain countries have defined that all data must reside within their borders.

If you are a global organization and have users around the globe and have transactions from multiple countries where the data must reside in the country where the data was transacted, this is another great option. This is where having the geographic distribution feature is valuable.

Performance

Along the same lines as the example above with having a global organization, there could be users who want to access their reports with the best performance possible.

The business users could be on the other side of the world, so when one part of the business has finished their working day, the other part of the business is just starting their day. Or it could be where one user is in Europe and another user is in Southern Africa.

By having the capability to be able to have the data hosted in the same country or close by a location means that interacting with the reports is a lot faster. This is because potentially the data does not have to travel half way around the world to be executed, and then travel half way around the world getting back to see the results.

As shown below for data to travel from the USA (New York) to the UK (London) is roughly 5,500km and would take an estimated 200 milliseconds (ms) to travel there.

Whilst if the data had to travel within the UK, it would travel possibly less than 100km and would take less than an estimated 40ms

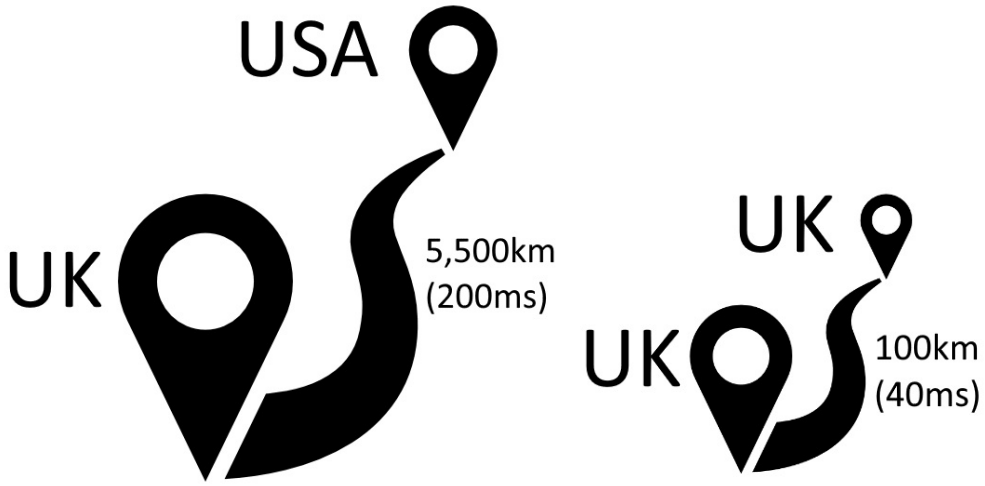


Figure 16-11: Distance between 2 offices.

Another advantage of looking to use geographic distribution is that if there is any maintenance or updates required to reports, they could be done in an App Workspace whilst those users are not at work. This allows for additional testing and ensuring that the reports and dashboards are updated and ready for those users when they get into the office the following day.

Dataflows

Dataflows are in Power BI is a way to unify data from different sources using Power Query Online. All the data is stored within Power BI (Azure Data Lake Gen2) and allows other users to consume dataflows within their reporting.

Below some considerations where using dataflows in Power BI premium might help assist you in moving some capabilities to Power BI Premium.

Linked Entities

It is best to explain what linked entries are in dataflows.

Let me take a step back and explain when you import data using dataflows, the data is taken from the source and then put into a dataflow.

Now each time you refresh data in a dataflow, it then goes back to the source and refreshes the data from the source.

Now when the linked entity is refreshed, it does not have to go back to the source data, but rather it gets the data from the existing dataflow. This allows for a much faster refresh of the data because it simply gets the data from the dataflow that is already stored in the Power BI Service.

One of the key features of a linked entity is that as a business user, you do not have to understand how the linked entities pieced (how they relate to each other) together, and in what order they need to be refreshed.

All of this is handled by Power BI Premium, and if there is an underlying dataflow that must be refreshed for your linked entity to be refreshed, Power BI Premium is aware of this and will do it for you automatically.

Computed Entities

Another feature that can assist an organization is computed entities.

Where computed entities it is different is that you created a linked entity off an existing dataflow. It also allows for you to create in-storage computations. In-Storage computation means that it does not have to go back to the source data to complete the computations, but rather, it is done within the existing dataflow.

It could also be described as when using the Reference, Duplication, Append or Merge functions in Power Query.

These computed entities can then be linked to other entities that you have already created.

Computed entities also mean that when the data is refreshed, it is then taken

from data that already exists in the dataflow. This speeds up the dataflow refresh performance significantly.

Incremental refresh

As explained previously on how incremental refresh works and what it does.

There is also the capability to use incremental refreshing in dataflows. This has the same advantages in that it does not have to refresh the entire dataflow.

As well as also allow for consuming fewer resources and a faster refresh of the data.

Parallel Execution of transformations

Another reason to potentially investigate using dataflows in Power BI Premium is that it allows you to have your transformations run in parallel.

This means if you have multiple dataflows refreshes happening at once, more than one can run at the same time. Which once again increases the performance of the dataflow refresh.

Enhanced Compute Engine

One of the new features that are coming to Power BI Premium is an enhanced compute engine.

This allows you to be able to ingest data up to 20x quicker than is currently possible. This is done by using a new compute layer which can ingest the data in a very efficient manner. It could be described as almost SQL like when ingesting the data. This makes it extremely efficient when ingesting data that requires joins or merges.

Ideally, you would import your data using the new compute engine and then use the computed entities to complete the in-storage computations as explained in the section above on how to leverage computed entities.

Monitoring for Power BI Premium

Whilst this does not cover when to move to Power BI Premium specifically, ensuring that you monitor your Power BI Premium capacity will enable that you are using all the resources available to you.

Along with this, if you start running into performance issues, having the ability to use the Power BI Premium monitoring will allow you to quickly and easily troubleshoot where there is a performance issue.

There are the following workloads that are currently available in Power BI Premium (as at July 2019)

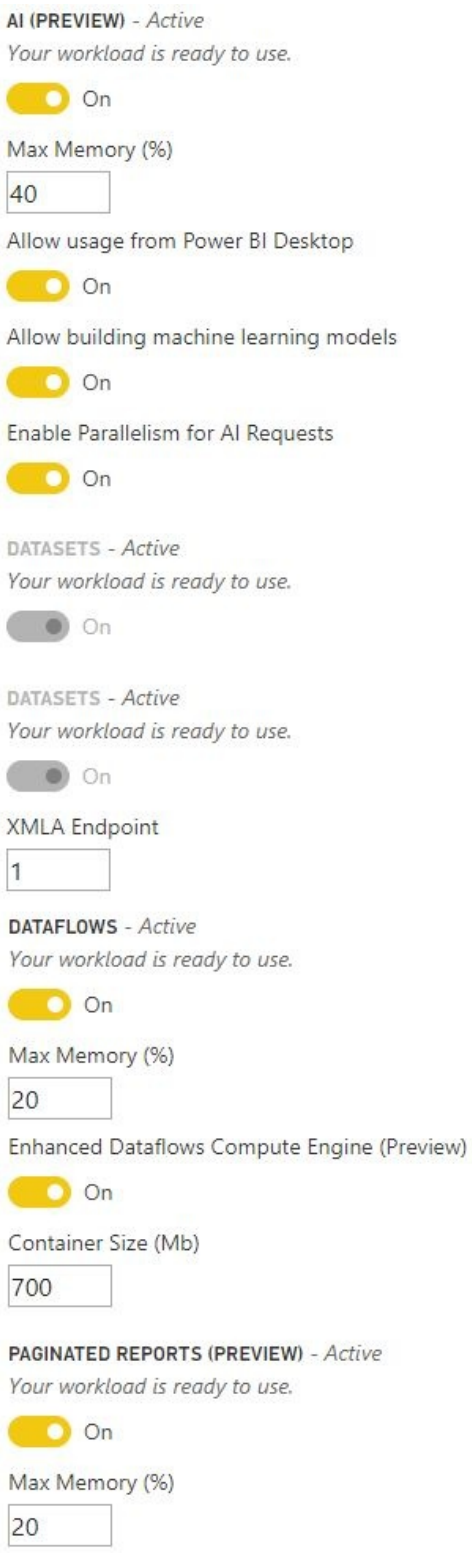


Figure 16-13: Power BI Premium Capacity Settings

There is also an insightful and detailed Power BI Premium performance

whitepaper that I would recommend any user who is administering Power BI Premium read.

The link to Power BI Premium can be found here: https://docs.microsoft.com/en-us/power-bi/whitepaper-powerbi-premium-deployment

Summary

There are multiple considerations that you must investigate when moving to Power BI Premium.

In this chapter, I have gone through the different Power BI capabilities and by gaining a better understanding of how each of the capabilities works.

AI is becoming essential in a lot of organizations, and when looking to use Power BI Premium, there are more AI features being released, which allows for easier integration with your existing data assets, which allows organizations to quickly use AI for better data-driven decisions.

With this knowledge of the capabilities, you can then leverage the Power BI Premium capabilities to enhance further or solve future or current reporting requirements in Power BI Premium.

About the Author

I am a Power BI & Data Analytics Consultant, having over 12 years' experience working in Business Intelligence or Data Analytic solutions on the Microsoft Platform.

I have successfully implemented solutions for small to large enterprise customers.

Recently I have focused on Power BI, with its rapidly changing features and incredible uptake it has allowed me to consult in a variety of different and challenging solutions.

I was awarded the Microsoft MVP award for Power BI since 2017.

I have spoken at the Microsoft Business Applications Summit (Seattle & Atlanta), Power BI World Tour & SQL Saturdays.

Proven competencies in the implementation of data analytic solutions from the ground up. Which included developing data warehouses, SSAS Cubes, and most recently Power BI solutions for customers in various business sectors.

I have worked within teams, managed teams as well as worked alone on various successful data analytics projects.

Chapter 17: Incremental refresh

Author: Michael Johnson

Abstract: When it comes to managing loading data, Power BI does not provide many options as to how data is loaded into the model, and this causes issues in larger models where full refreshed as time-consuming. Incremental refresh offers a simplified approach to loading only the most recent changes to the data making data loads faster and more reliable while consuming fewer resources. In this chapter, we cover how to set up and maintain Incremental-refresh for your power BI model.

Introduction

Working with large volumes of data within Power BI can often be very time consuming, especially when it comes to refreshing datasets and it is not uncommon for large datasets refreshes to take several hours. In a world that demands near real-time insights, these loads have become an impediment to the business.

Incremental Refresh was added to the Power BI service to reduce the time taken to refresh a dataset by merely refreshing only the most recent data and not the entire dataset. By reducing the amount of data that needs to be re-loaded, refreshes take not only less time but also consume fewer resources doing it.

What is incremental refresh and how does it work

Incremental Refresh implements partitioning where a partition is a small part of a larger table, each partition is assigned a range of values usually by the date that gets stored in that partition. The power of partitioning is that the Power BI service can refresh a single partition or even a group of partitions instead of the entire table as we see in Power BI service currently. It is generally true that our older data never changes such as last year`s sales and then new data is continuously changing, such as today`s sales. The ability to refresh only today`s sales and not last year’s sales results in a lot of saving of time and effort. Thus, benefits of this incremental refresh are:

Faster Refreshes:

As each partition contains only a small percentage of the total rows, we reduce the number of rows that need to be imported.

Fewer resource required

As only a small portion of the data is loaded the number of resources required from both the source system and the power BI service is reduced including locks on source systems, CPU and network resources.

More reliable processes

As Power BI datasets often use core business systems as their source, long running queries may affect their performance. By reducing the number of rows, we improve not only the speed of the load but also reduce the risk of query timeouts and other network related issues.

Requirements

Before setting up Incremental Refresh there are a few requirements that must be met

Premium Workspace

Incremental refresh is currently only available to datasets published to a premium workspace, while the Incremental Refresh policy is set up in Power BI Desktop the feature is not activated until the report is published to a premium Workspace. The report can also not be published to a non-premium workspace such as ‘My Workspace’ once it has an Incremental Refresh policy applied.

Query Folding data source

While not absolutely necessary, it is recommended that Incremental Refresh only be set up on a data-sources that support query folding. Query folding is the ability of Power Query to adapt some of the transformations that are usually done in the mashup engine and push then down to the source system. Many sources, such as text files and websites, are unable to do this. However, data sources such as relational database and some OData sources can do this.

Transaction dates

For Incremental refresh to work, it requires a date of transaction (such as a sales or event date). This record should also be non-volatile meaning that details of this event should not change after a period of time as these changes will not be loaded if they occur outside of the refresh window. Another requirement for this date column is that there are no future dated transactions as these will not be loaded as part of the incremental refresh.

Enabled Incremental Refresh feature

At the time of writing Incremental refresh is still a preview feature. Therefore, you need to enable it in the Power BI preview features tab which can be found at

File ?? Option and Settings ?? Options ?? Preview features

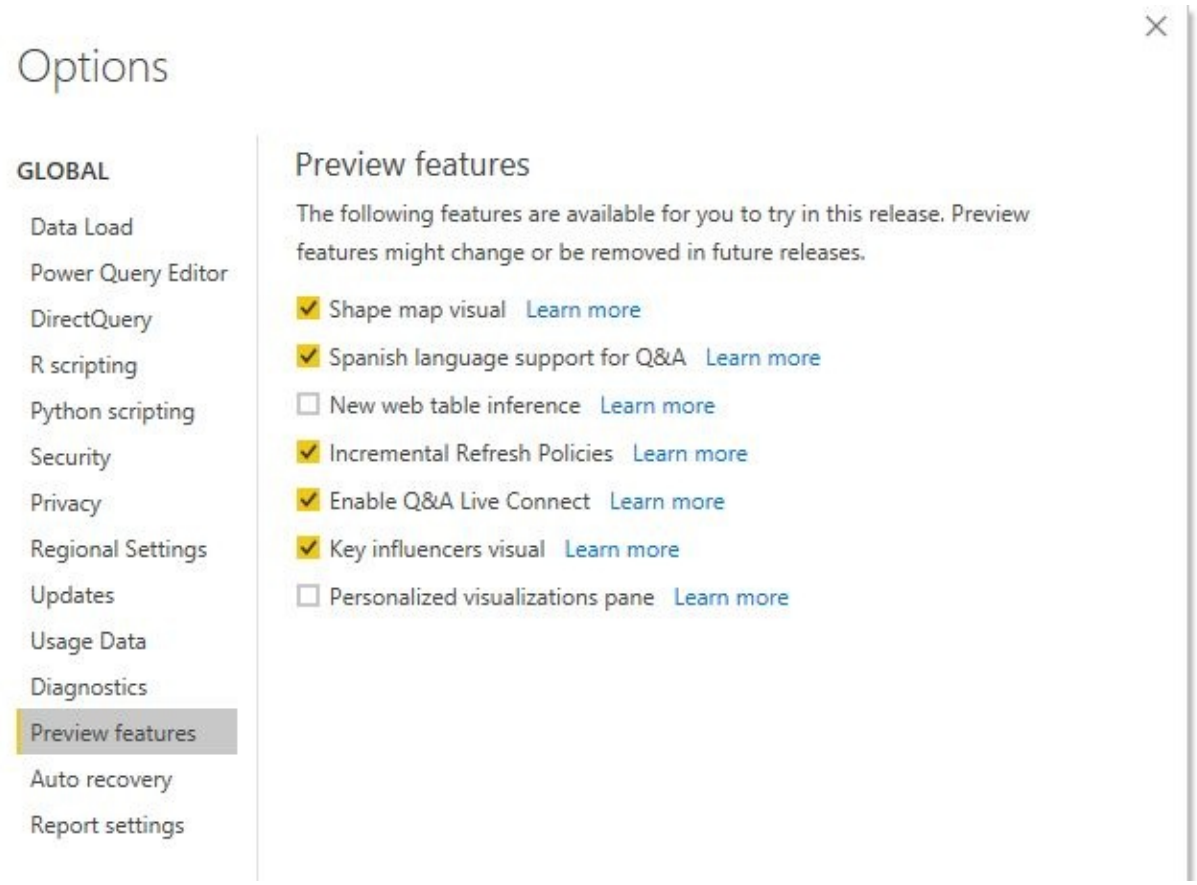


Figure 17-01: Enabling the Incremental Refresh preview feature

Once these requirements have been met, we can setup incremental refresh.

Setting up Incremental refresh

Setting up incremental refresh can be done in 4 easy steps

- **Step 1**: Create range parameters
- **Step 2**: Filter dataset using range parameters
- **Step 3**: Configure incremental refresh in Power BI desktop
- **Step 4**: Deploy and publish report

Step 1 – Create range parameters

For Incremental Refresh to be configured first need to create two specific parameters within the report called **RangeStart** and **RangeEnd**, these parameters must be defined as the Date\Time data type. The two parameters are used to determine the upper and lower boundary values of each partition, **RangeStart** being the first date in the partition and **RangeEnd** being the first date of next partition i.e. Up to but not including the **RangeEnd**. The selection of dates for these parameters is not critical and dates should be chosen to support the report development process such as a range of 13 months.

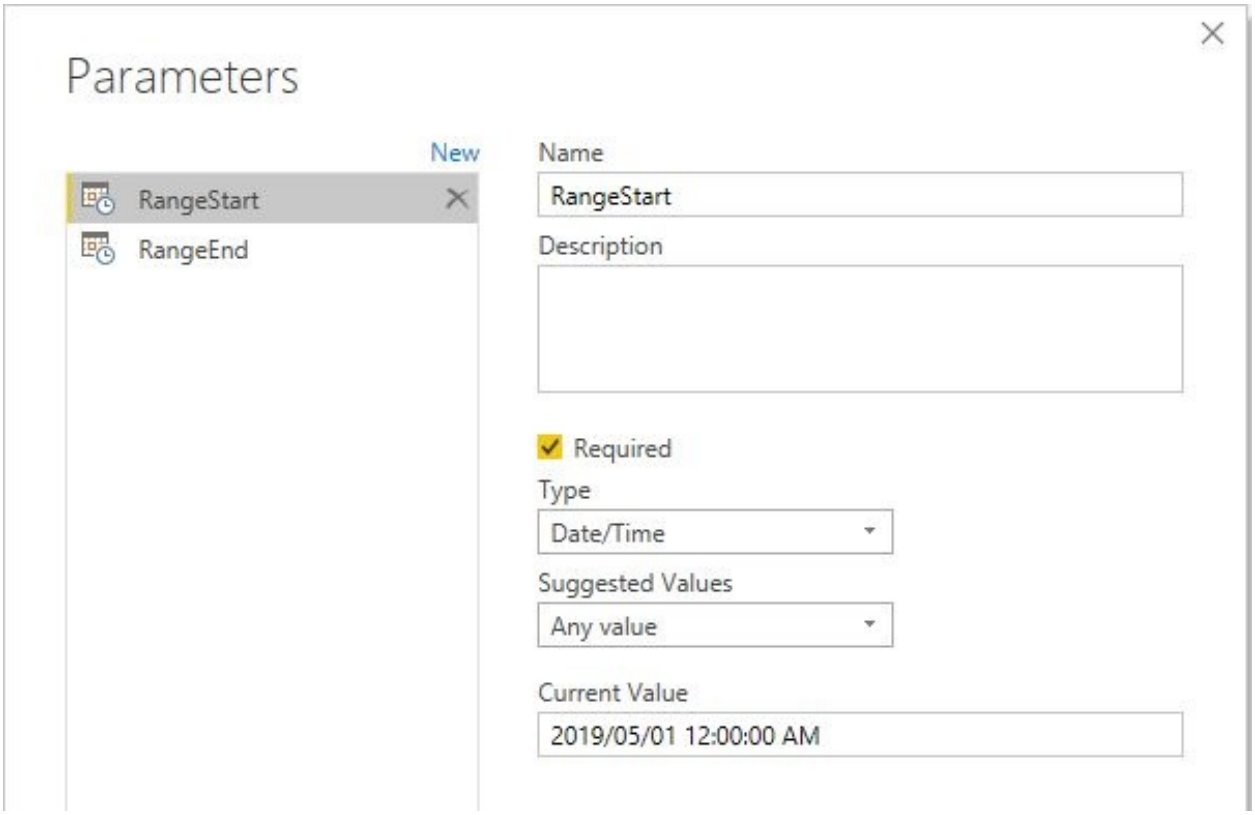


Figure 17-02: Adding the new RangeStart and RangeEnd parameters to report

Step 2: Filter dataset using parameters

Once the two required parameters have been created, the next step is to filter the large table using those parameters by limiting the range of data to only dates between the two values. Once published Power BI dynamically replaces these original parameter values with new upper and lower bound values for each partition.

The date filter can easily be applied by using the between operator to the date column.

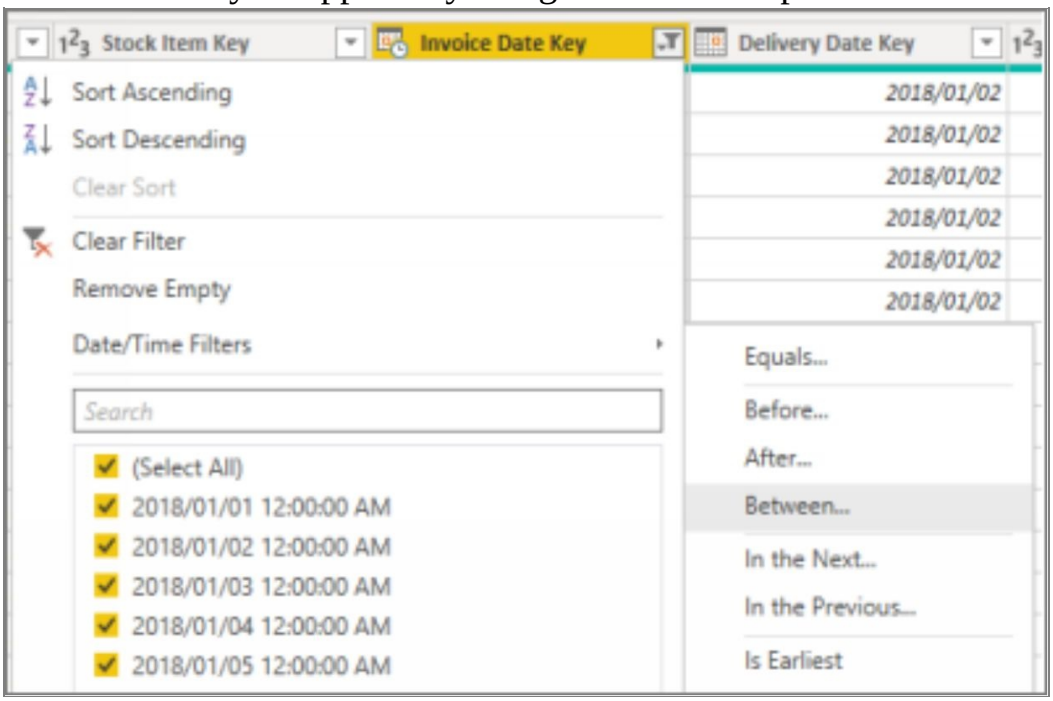


Figure 17-03: Adding a filter to the date column

Then the filter needs to be configured. Note: in order to avoid missing or duplicated records it is important that the ‘is after or equal to’ is selected for the first filter step. The next step is to assign this the parameter value by selecting the parameter option then selecting the **RangeStart** parameter, which is in the list of available parameters. The second filter is applied in the same way this time using the ‘is Before’ option and the **RangeEnd** value.

Figure 17-04: Assigning parameters to table filter

Alternatively, the filter can also be added by using M, The M code for this operation would look something like this

```
=  Table.SelectRows(#"Changed Type", each [Invoice Date Key] >= RangeStart and [Invoice Date Key] < RangeEnd)
```

Step 3: Configure incremental refresh in Power BI desktop

Once the table has been correctly filtered using Power Query the next step is to set up the Incremental Refresh policy. This is done in the Power BI editor (You should select Close and Apply from the Power Query editor to return). The incremental policy can be found on the context menu (Right click on the table name) for the for the table.

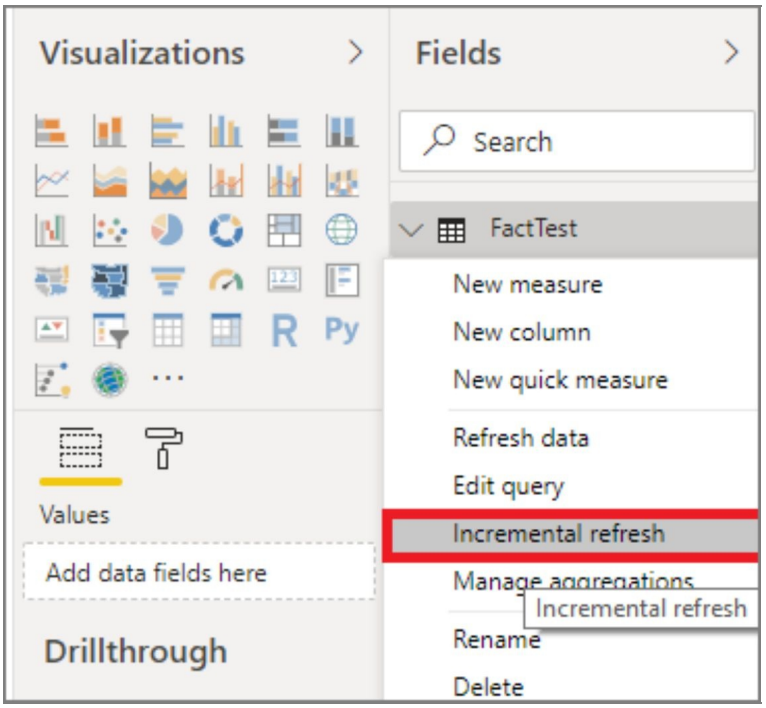


Figure 17-05: Adding Incremental refresh policy

Before the Incremental Refresh policy is set up, Power BI validates that the first two steps have been completed correctly, if there are any issues an error message such as the one below is displayed, this will need to be corrected before proceeding.

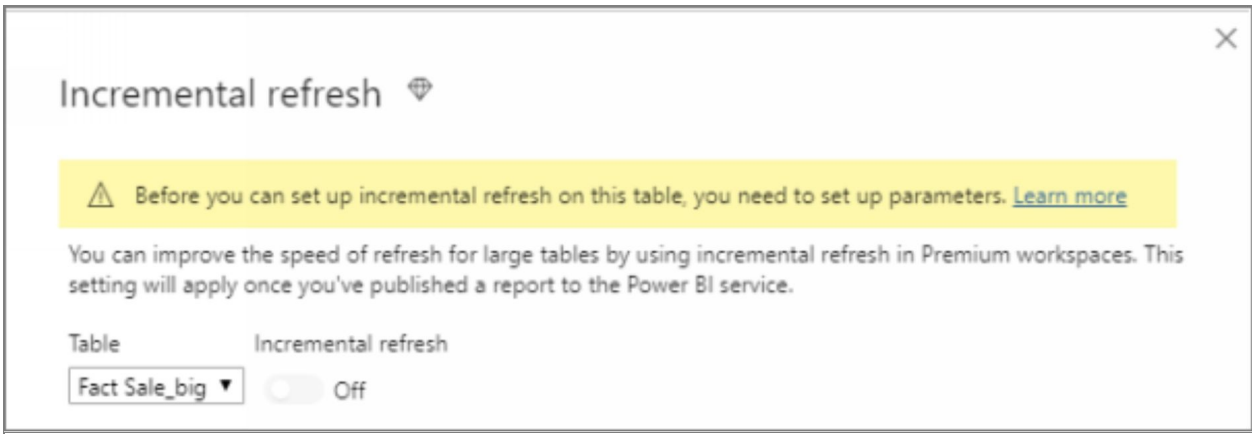


Figure 17-06: Incremental refresh unable to be setup due to error

If all the prior step were successfully completed, then the option to enable Incremental Refresh is provided by toggling the incremental tab alongside the name of the table that you want to configure incremental refresh.

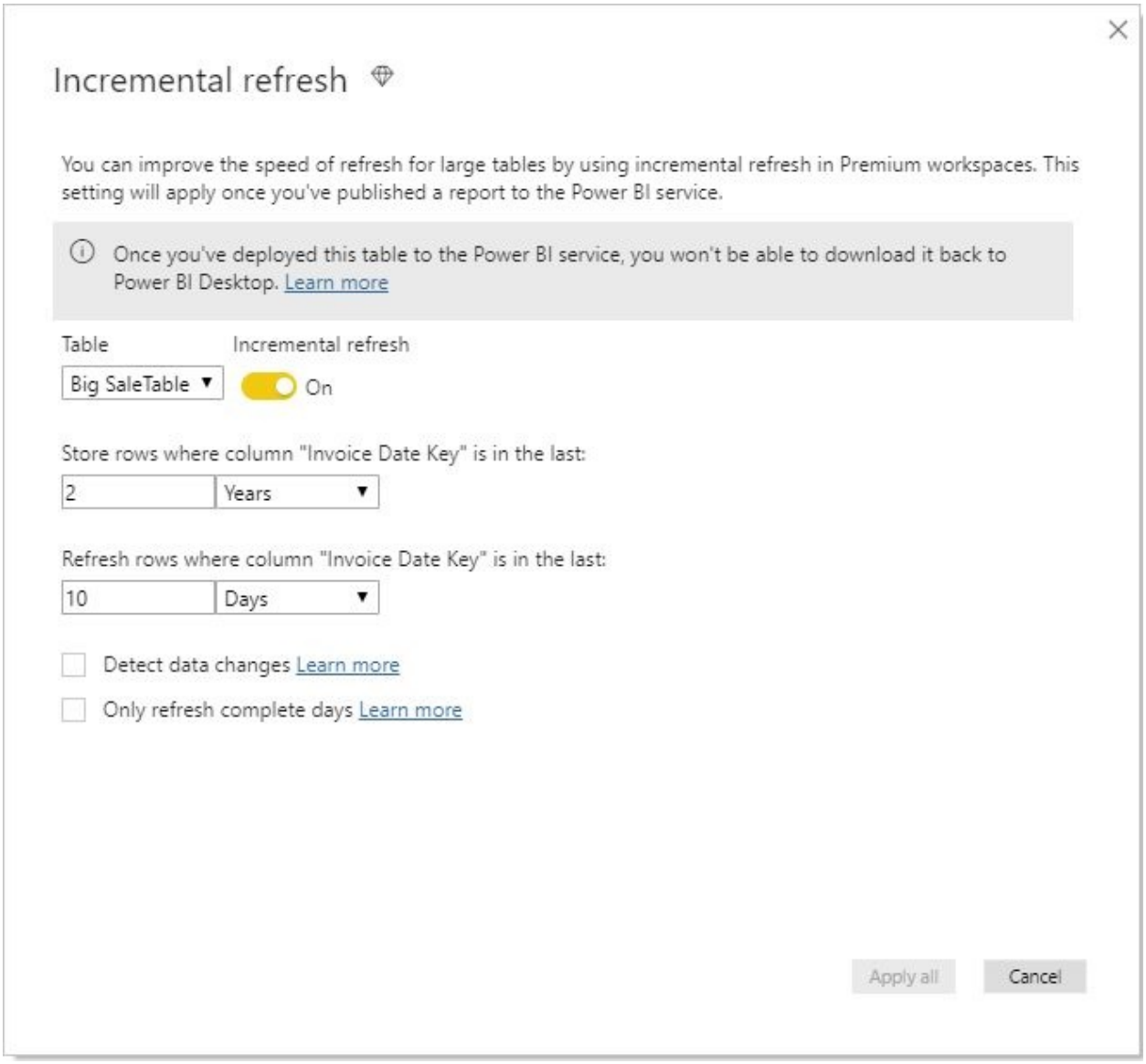


Figure 17-07: Configuring the Incremental refresh policy

With the incremental refresh option enabled there are 4 configurations that need to be set

Store rows where "Date column" is in the last: The first configuration defines how much data must be kept in the model. In the case of the policy above at least 2 years of data are stored.

Refresh rows where the column "Date column" is in the last: This configuration option sets the number of periods that will be loaded each time the