AI/ML Functionality in Arbutus Analyzer

ARBUTUS
Powerful Analytics Simplified

Contents

A | AI-ML Functionality - Introduction

The purpose of this section is to introduce you to some of the more common AI-ML Apps that can be found on the Apps menu. These include:

- Outliers (ML)
- Clusters (ML)
- Sentiment Analysis (ML)
- Smart Query (AI)
- Data Categorization (AI)
- Arbutus Assistant (AI)

Each of the above AI-ML Apps are in fact Arbutus procedures. When run, they call a specific Python procedure, e.g., **SentimentIntensityAnalyzer,** that uses AI-ML technology to perform the actual testing and generate the desired results, which are then passed back to Arbutus via the same Arbutus procedure.

Arbutus employs AI and machine learning algorithms to strengthen your ability to analyze vast amounts of data efficiently. These algorithms can identify patterns, trends, and outliers that may indicate potential risks or anomalies within the data.

IMPORTANT – PLEASE NOTE

1. All ML capabilities are included in Arbutus Procedures as a bundled Python environment and therefore all data and data processing remain on customer premises.
2. Please note that our AI functionality requires internet connectivity and an active ChatGPT account. Otherwise, it is disabled and non-functional by default, to ensures that no data is sent off premise unless explicitly decided upon by the customer's organization and users. It will only be enabled once configured with your organization's OpenAI account details. Please take note that the AI Apps will use any information you share via the input prompts or any data from a table that you specify/select, and this information/data is then passed on to OpenAI (via ChatGPT) for subsequent analysis. Therefore, it is imperative that you get confirmation on your organization's policy allowing the use of ChatGPT.

   *For more information on configuring Arbutus to work with your ChatGPT/OpenAI account, please see the section on **Requirements for using AI Smart Apps in Analyzer** on page 7 of the Arbutus Analyzer Install Guide. The Install Guide is available for download on the Arbutus website at this link.*

The following seven sections cover these AI-ML Apps:

1. **Section A** – AI-ML Functionality: Introduction
2. **Section B** – ML Functionality: Outliers
3. **Section C** – ML Functionality: Clusters
4. **Section D** – ML Functionality: Sentiment Analysis
5. **Section E** – AI Functionality: Smart Query
6. **Section F** – AI Functionality: Data Categorization
7. **Section G** – AI Functionality: Arbutus Assistant

B | ML Functionality: Outliers

What are outliers?

Outliers, in the context of information evaluation, are records with numeric amounts that deviate or differ significantly from the numeric amounts in the records that they are grouped with. These anomalies can show up as surprisingly or unexpectedly high or low values, thus disrupting the distribution of data.

For example, in a dataset consisting of weekly sales figures, if the sales revenue for one week is extensively higher than the sales revenue for all the other weeks, then that sales revenue would be considered as an outlier.

Using numerical values as an example, suppose that in an Accounts Payable file, invoices from a particular vendor typically range between $2,000 and $5,000. However, you find one invoice for $11,000. This would be considered an outlier, though further reviews may be required to confirm that.

It is also important to keep in mind that a record can be an outlier for a perfectly legitimate reason. This is why you should perform additional reviews of the outliers identified in your dataset to confirm that they in fact exist.

Why is dealing with and/or removing outliers necessary?

- Removing outliers can help ensure the analysis is based totally on a more representative sample of the dataset and the information it represents.
- Dealing with outliers is critical in data analysis to ensure the integrity and accuracy of the results – they can significantly skew your analysis and lead to misleading conclusions.
- Materiality of a few large questionable transactions can lead to misstatements in financial statements.

Causes of outliers

Outliers can arise from various sources. Here are some examples:

- **Data entry errors / human errors** – data incorrectly entered manually can result in extreme values, high or low
- **Intentional outliers / fraud or malicious activity** – in instances of fraud, where data might be manipulated deliberately
- **Data processing / conversion errors** – issues during data cleansing, transformation, or integration can introduce outliers
- **Outdated data current data** – using outdated or obsolete data can result in outliers when compared to current dataset

continued...

What are the different statistical methods and key elements for identifying outliers?

There are different statistical methods used for identifying outliers. Here are three of them:

1. **Z-Score (standard score) method** – this method uses the Mean and Standard Deviation to measure how many standard deviations a data point is from the Mean. This method is generally used and suited for normally distributed populations.
2. **Modified Z-Score (using Median and Median Absolute Deviation (MAD)** – this method is more robust to skewed data than the standard Z-score. It uses the Median and the MAD and is better suited for non-normal distributed populations.
3. **Interquartile Range (IQR) Method** – this is based on quartiles, typically the 25th and 75th percentiles in the dataset. It is more robust than standard deviation-based methods.

Whichever method is used, there are certain key statistical elements that are used in algorithms or routines to identify outliers. The Standard Deviation is one of these key elements. Other key elements can include the Mean, Median, and Inter Quartile Range (IQR) percentiles. Typically, as a result of these algorithms or routines, a lower and upper boundary is established. Any numerical amount that does not fall within the lower and upper boundary would then be flagged as outliers in the dataset tested.

Using Arbutus technology to identify outliers

Arbutus offers two capabilities for identifying outliers in your dataset.

1. As part of ML functionality, leveraging Python technology by using an industry-standard Python procedure
   - Go to Apps > AI-ML > ML02 Outliers

   This functionality uses the Inter Quartile Range methodology.
2. Using a procedure, well established over time, deployed in Arbutus technology
   - Go to Apps > Smart Apps > Miscellaneous > MI01 Outliers

   This functionality uses the Modified Z-Score methodology (uses the Median), which is better suited for non-normal distributed populations.

   Note: This section also includes the key steps in developing a procedure on your own to follow the Z-Score methodology (uses the Mean) instead, which is better suited for normally distributed populations.

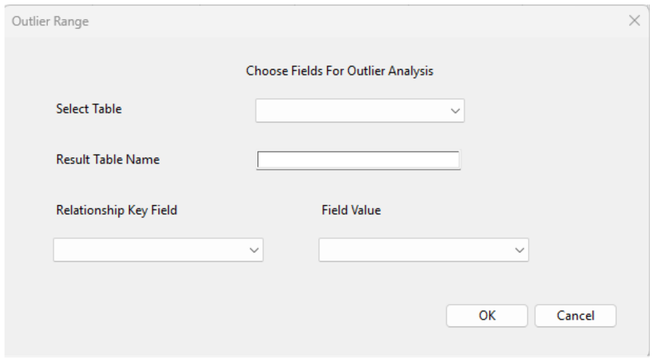
continued...

Using the Arbutus ML functionality

The Arbutus ML Outliers functionality involves executing an industry-standard Python procedure **(iqr.py)** designed for identifying outliers using Interquartile Range methodology (IQR), thus calculating the medians and the lower and upper boundaries for identifying outliers in a dataset.

The ML Outliers App can be accessed and run from the **Apps** Menu: **Apps > AI-ML > ML02 Outliers.**

When you run this app, the **Outlier Range** dialog is displayed.



The **Outlier Range** dialog has the following input prompts for you to complete.

1. **Select Table** – this is a dropdown of all the tables in the current Analyzer project. Select the table on which you would like to run Outliers.
2. **Results Table Name** – this is a text box for you to enter the name you would like to give the results/output table. Enter the name in this text box.
3. **Relationship Key Field** – this dropdown requires you to select the key field on which the records are grouped by in your dataset, e.g., Vendor ID, City, etc. If your dataset does not contain such a field, then you can also select a different field that is unique, e.g., a field with a value that is unique throughout the entire data set.
4. **Field Value** – this is a dropdown of all the numeric fields in the table that you would have selected earlier from the Select Table dropdown. The numeric field that you select from the Field Value dropdown is the field on which you would like to identify outliers on.

Output / Results

Upon completion of the Outliers procedure, an output table (that was specified earlier when you ran Outliers) opens containing the results of the outliers identified. In this output table, you will see a new field/column, called **outlier.** The value in this new field will either be **True** (implying an outlier) or **False**. The remaining fields in the output table are the fields from the original table that you would have selected when your ran Outliers.

The order of the records in the output table are arranged based on the field you had selected earlier from the **Relationship Key Field** dropdown. The **True** or **False** response is based on the numeric field you had selected earlier from the **Field Value** dropdown. The outlier is based on this field when compared to the other values in this same field from the records within the same **'Relationship Key Field'** grouping. To identify the other values from the same grouping that are being compared, it is recommended that you open the source table and look at the values in the field on which you ran outliers.

continued...

Using the Procedure deployed in Arbutus technology

To identify outliers in your dataset using the Arbutus procedure, follow these steps:

1. Go to Apps > Smart Apps > Miscellaneous and then click on MI01 Outliers.

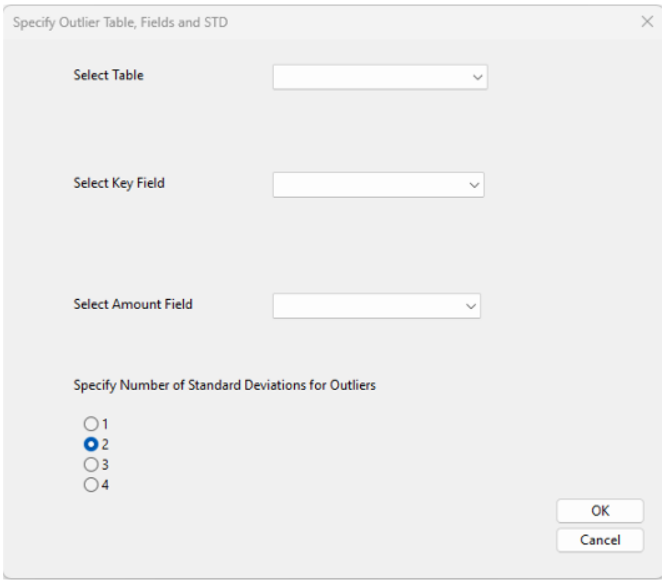
   The Outliers (Specify Outliers Table, Fields and STD) dialog is displayed. It includes the following input prompts:

   a. **Select Table** – click the dropdown to select your table

   b. **Select Key Field** – click the dropdown to select the key field from the selected table – see section below for more information

   c. **Select Amount Field** – click the dropdown to select the numeric (or amount) field you wish to test for outliers

   d. **Specify number of standard deviations** – click the radio button to specify the number of standard deviations. It defaults to 2 – in a normally distributed population, what is considered an outlier is typically 2 standard deviations and above. See the section "Specifying the number of standard deviations" below for more information. *Note: Step #2 follows later*


Specify Outlier Table, Fields and STD

Select Table

Select Key Field

Select Amount Field

Specify Number of Standard Deviations for Outliers

1 [✓ 2] 3 4

Selecting the key field

The key field that you select does not necessarily have to be a field whose records are grouped on. For example, you have records grouped by a City and you have a Sales field containing an amount for each of those cities.

When testing for outliers, you do not need a dataset containing grouped records. In fact, your dataset may not even have grouped records. You can still test for outliers across the entire dataset.

However, it is important to note that the Arbutus procedure does run a summarize based on the key field that you select. This also implies that any outliers identified are those that belong to a group of records – that's assuming if a group exist, otherwise it is based on the entire dataset.

More information on how the summarize command is used in this procedure is included later in this section.

continued...

Understanding statistical values obtained from the Summarize command

When identifying for outliers, the Arbutus procedure uses the Summarize command to obtain the following key statistical values:

a. **The standard deviation** – this is a is a measure of how spread out the numbers in a data set are. It tells us how much the numbers in the data set differ from the average (mean) of the data set. Standard deviation is typically derived from the Mean (Average) – see #b below.

b. **The Mean (Average)** – this is the average value of a list of numeric values (basically summing the numeric values and dividing the sum by the number of numeric values).

   As mentioned above, the standard deviation is typically derived from the mean (rather than the median). The mean is used as a central point for calculating how much each data point deviates from it. As well, the using the mean is better suited for datasets where there is a normal distribution of the population. As such, in cases where the dataset is heavily skewed and contains known outliers, the median might be a better measure of central tendency. For skewed distributions, other measures like the Interquartile range (IQR), as used in the Python procedure technology, might be more appropriate for understanding variability.

c. **The Median** – this is the "middle" value in a list of numeric values. For example, in a list of five numbers 5, 12, 14, 79, 90, the median is the "middle" number, which is 14.

   What if we took the average (mean) of the five numbers? The total is 200. Given that we have 5 numbers, the average (mean) is 40. Now compare that with the median, which is 14. The difference is indeed noticeable!

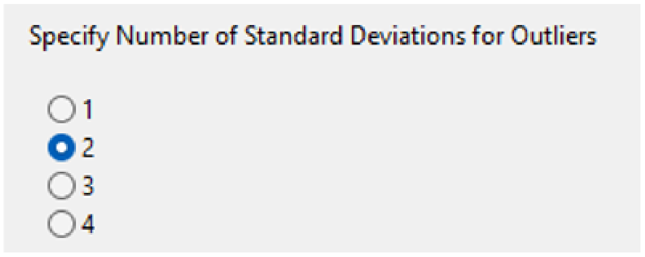
d. **The Q1 and Q3 quartiles**

   Q1 – this is the first quartile and is determined from the lower half of a numeric data set that are to the left of the median (middle) value when the data has been put into increasing/sorted order. The first quartile, denoted by Q1, is the median of the lower half of the data set. This means that about 25% of the numbers in the data set lie below Q1 and about 75% lie above Q1

   Q3 – this is the third quartile is determined from the upper half of a numeric data set that are to the right of the median (middle) value when the data has been put into increasing/sorted order. The third quartile, denoted by Q3, is the median of the upper half of the data set. This means that about 75% of the numbers in the data set lie below Q3 and about 25% lie above Q3

continued...

Specifying the number of standard deviations


Specify Number of Standard Deviations for Outliers

1 [✓ 2] 3 4

Why is there a need to specify the number of standard deviations, and why is it defaulting to **2** in the **Outliers** dialog?

It is understood that in a normal distributed population, what is considered an outlier is **2 standard deviations and above.**

For the same dataset, **as you increase the standard deviation multiple** (the number of standard deviations), you potentially **decrease the number of outliers** in the output results. This is illustrated below.

The two charts below demonstrate that increasing the number of standard deviations can potentially reduce the number of reported outliers. To begin, suppose that you have the following numbers in your dataset: -5, -4, -2, 2, 4, 5, 7, 7, 9, 12. The mean (average) of the numbers is 3.50. The calculated standard deviation is 5.39.

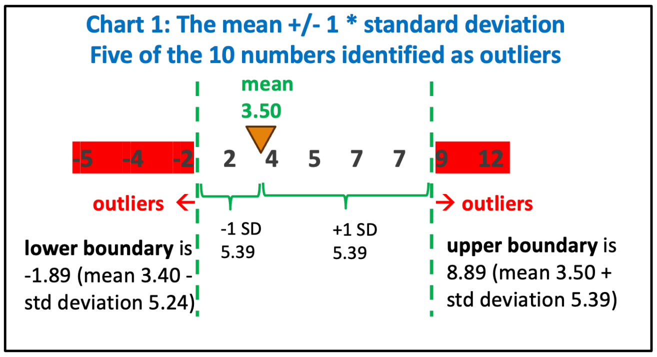


Chart 1: The mean +/- 1 * standard deviation
Five of the 10 numbers identified as outliers
mean
3.50
-5 -4 -2 2 4 5 7 7 9 12
outliers
outliers
-1 SD
5.39
+1 SD
5.39
lower boundary is -1.89 (mean 3.40 - std deviation 5.24)
upper boundary is 8.89 (mean 3.50 + std deviation 5.39)

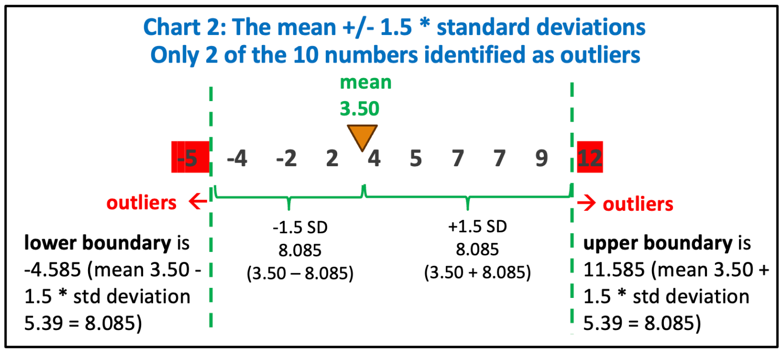


Chart 2: The mean +/- 1.5 * standard deviations
Only 2 of the 10 numbers identified as outliers
mean
3.50
-5 -4 -2 2 4 5 7 7 9 12
outliers
outliers
-1.5 SD
8.085
(3.50 – 8.085)
+1.5 SD
8.085
(3.50 + 8.085)
lower boundary is -4.585 (mean 3.50 - 1.5 * std deviation 5.39 = 8.085)
upper boundary is 11.585 (mean 3.50 + 1.5 * std deviation 5.39 = 8.085)

2. Once you have selected the table, the key field, the amount field, and specified the number of standard deviations in the Outliers dialog, click OK to continue.

Output / Results

Upon completion of processing, the Arbutus procedure will display a pop-up message notifying you of the number of outliers that were identified, if any. Otherwise, the pop-up message will simply say that no outliers were found.

A new table, MI01 Outliers, is created in your Analyzer project containing all records identified as outliers based on the key (amount) field tested/analyzed.

Note: The Arbutus procedure extracts outlier records based on the lower and upper boundaries. While outliers below the lower boundary might be seen as less significant in terms of materiality compared to those above the upper boundary, it is still beneficial to examine them. However, manual review should primarily focus on the outliers above the upper boundary.

continued...

Key processes in the Arbutus procedure

Outlined below are the high-level steps for developing a procedure to test for outliers, akin to how the current Arbutus Outliers procedure works.

1. Open a table. It can be a table where records are grouped or not grouped at all. However, it must contain a numeric field, such as an amount. For illustration purposes, let's call this **Table_A_Orig.**

    ```
    OPEN Table_A_Orig
    ```

2. Use the Summarize command to summarize on a key field, e.g., VendorID, and specify five different Types on an amount field, e.g., SalePrice, to process on. Let's name the output table **Table_B_Summ.**

    ```
    SUMMARIZE ON Vendor_ID FIELDS SalePrice SUMMTYPE Stddev SalePrice
    SUMMTYPE Avg SalePrice SUMMTYPE Median SalePrice SUMMTYPE Q1 SalePrice
    SUMMTYPE Q3 TO "Table_B_Summ" PRESORT
    ```

    *Note: The field **SalePrice** is specified multiple times with the **FIELDS** parameter, each time as a different type, **SUMMTYPE***

3. Open the resulting summarized file as a Secondary file in preparation for a 'join' back with the source file.

    ```
    OPEN Table_B_Summ SECONDARY
    ```

4. Join the source file with the resulting summarized file using **Vendor_ID** as the key field from both the files. Name the output table **Table_C_Join**.

    ```
    JOIN PKEY Vendor_ID FIELDS ALL SKEY Vendor_ID WITH Stddev_SalePrice Avg_
    SalePrice Median_SalePrice Q1_SalePrice Q3_SalePrice TO "Table_C_Join"
    PRESORT SECSORT
    ```

5. Close **Table_A_Summ** as the secondary table

    ```
    CLOSE SECONDARY
    ```

6. Open the resulting joined file and create a conditional computed field that will display a message to indicate whether or not an outlier in a record was found. Let's call this conditional computed field **c_Outliers_Exceptions**

    ```
    OPEN TABLE_C_JOIN

    DELETE FIELD c_Outliers_Exceptions OK
    DEFINE FIELD c_Outliers_Exceptions COMPUTED

    "Outlier"              IF SalePrice < (Avg_SalePrice – (2 * Stddev_
    SalePrice)) OR SalePrice > (Avg_SalePrice + (2 * Stddev_SalePrice))
    "Not an Outlier"
    ```

    Note: In the conditional computed field expression:

    a. We hardcoded the standard deviation multiple to 2 – this value represents the number of standard deviations., e.g., two times the standard deviation. Also, we are specifying 2 because in a normally distributed population, what is typically considered as outliers is two standard deviations and above.

continued...

b. In an appropriate place in your procedure, you can also use a variable to store this value. For example:

```
v_stddev = 2
```

c. If you do use a variable like the one listed above, you can then replace the hardcoded value of 2 in the expression of your conditional computed field. For example, replace:

```
(2 * Stddev_SalePrice) WITH (v_stddev * Stddev_SalePrice)
```

7. Create a new extract to include the results. Let's call this resulting file Table_D_Results

```
EXTRACT FIELDS ALL TO "Table_D_Results"
```

Enhancing the Arbutus procedure

You can enhance your procedure by:

1. Incorporating a DIALOG (command) in your procedure to prompt for:

   a. Source table – dropdown selection listing all the tables in the current Analyzer project
   b. Field to group outliers on – dropdown selection listing all the character fields in the current Analyzer project
   c. Numeric field to test outliers on – dropdown selection listing all the numeric fields in the current Analyzer project
   d. Number of standard deviations – radio button selection
   e. Name of final resulting table

2. Using macro-substitution on variables specified in the DIALOG command in subsequent commands in the procedure, e.g., Summarize, Join. For example, suppose that in the DIALOG command you assigned variables as follows:

   a. Field to group outliers on: v_key
   b. Numeric field to test outliers on: v_amount

   Then, in your Summarize command, you could simply say: SUMMARIZE ON %v_key% FIELDS %v_amount% SUMMTYPE…

   And in your Join command, you could say: JOIN PKEY %v_key% FIELDS …

3. Modifying your conditional computed field to break down the outliers into two new categories, rather than just "Outlier" and "Not an Outlier"

   a. "Outlier – below Lower Boundary" IF SalePrice < (Avg_SalePrice – (2 * Stddev_SalePrice)

   Or if you are using macro-substitution, then:
   *"Outlier – below Lower Boundary" IF %v_amount% < (Avg_SalePrice – (v_stddev * Stddev_SalePrice)*

   b. "Outlier – above Upper Boundary" IF SalePrice > (Avg_SalePrice + (2 * Stddev_SalePrice)
   c. Your default value could just be the same as before, "Not an Outlier"

continued...

C | ML Functionality: Clusters

What are clusters and data clustering?

In machine learning, clusters and clustering refer to groups of data points that are similar to each other. For example, in real estate, neighbourhoods with similar property values, style, or demographics could form a cluster. Similarly, other clusters are also formed.

Clustering is the process of identifying and grouping these similar items. It is a common technique in data analysis, machine learning, and market segmentation. Clustering helps in recognizing patterns, organizing data, and making predictions. For example, in real estate, clustering could be used to categorize buyers into groups based on budget, location preference, or property type, allowing for better-targeted marketing strategies.

Common clustering algorithms include K-means, Hierarchical Clustering, and DBSCAN.

The Arbutus ML Clusters App is an Arbutus procedure that in turn calls a Python 'k-means methodology' procedure **(kmeans.py)**.

High-level illustration of clusters and clustering of data

Consider a dataset with ten records, each having an amount field and other fields such as transaction ID and category, as shown in the table below.

Txn ID	Amount ($)	Category
1	100.00	Groceries
2	120.00	Groceries
3	110.00	Groceries
4	105.00	Groceries
5	150.00	Electronics
6	130.00	Electronics
7	115.00	Groceries
8	125.00	Electronics
9	2,500.00	Luxury
10	2,700.00	Luxury

Clustering process

1. **Data collection and preparation:** Gather transactional data, including the amount, category, and other relevant fields.
2. **Applying clustering algorithm:** Use a method like K-Means to group similar transactions based on amount field.
3. **Assigning clusters:** Label each transaction according to its cluster, e.g., low, medium, high spenders.
4. **Analyze clusters and take action:** Interpret the results to find insights and use findings for decision-making.

continued...

Based on the dataset used in our example, the clusters are identified as follows:

Cluster	Transactions included (ID #)	Amount Range
Cluster 1 (Low spenders)	1, 2, 3, 4, 7	100 – 120
Cluster 2 (Medium spenders)	5, 6, 8	125 – 150
Cluster 3 (High spenders)	9, 10	2500 - 2700

The final output after clustering would look like the table shown on the right:

Txn ID	Amount ($)	Category	Cluster
1	100.00	Groceries	Cluster 1
2	120.00	Groceries	Cluster 1
3	110.00	Groceries	Cluster 1
4	105.00	Groceries	Cluster 1
5	150.00	Electronics	Cluster 2
6	130.00	Electronics	Cluster 2
7	115.00	Groceries	Cluster 1
8	125.00	Electronics	Cluster 2
9	2,500.00	Luxury	Cluster 3
10	2,700.00	Luxury	Cluster 3

How does identifying of clusters help in data analysis / Why is clustering useful?

Identifying clusters in data analysis helps with:

1. **Pattern recognition** – reveals hidden trends and relationships within the dataset.
2. **Segmentation** – Groups customers, transactions, or properties based on similar characteristics for targeted strategies.
3. **Anomaly detection** – Identifies outliers or unusual patterns, useful for fraud detection.
4. **Decision making** – Supports better marketing, pricing, and investment strategies by understanding different groups within the dataset.
5. **Efficiency** – Simplifies large datasets by organizing them into manageable groups, making analysis more actionable.

Examples of clustering in real life

Here are some examples of clustering in real life

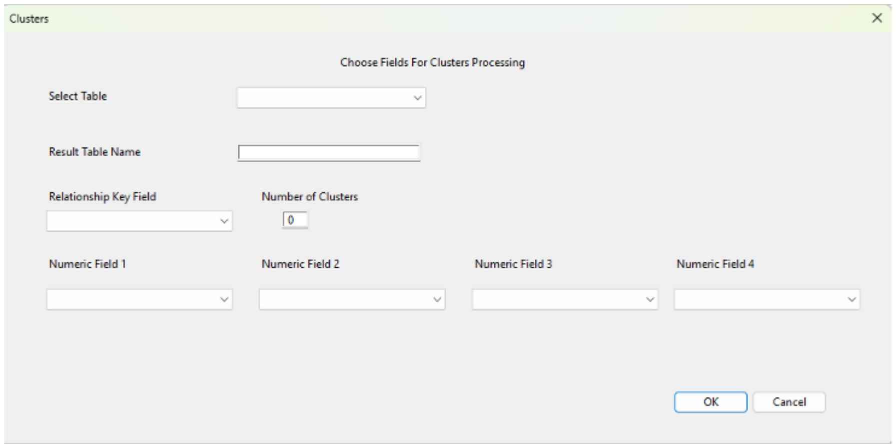
1. **Business & Marketing** – Customer segmentation for personalized marketing.
2. **Finance** – Fraud detection by identifying unusual transactions.
3. **Real Estate** – Grouping neighbourhoods by price range, property type, or buyer preference.
4. **Healthcare** – Identifying patient groups based on medical history.

continued...

Using the Arbutus ML Clusters App

The Arbutus ML Clusters App is an Arbutus procedure that issues a call to a Python procedure **(kmeans.py)** that applies the 'k-means' clustering algorithm and then assigns data points (from your dataset) to clusters. When you run the Arbutus procedure, you are prompted for the number of clusters to define.

The ML Clusters App can be accessed and run from the Tools Menu: **Apps > AI-ML > ML01 Clusters**



When you run this app, the Clusters dialog is displayed with the following input prompts for you to complete.

1. **Select Table** – this is a dropdown of all the tables in the current Analyzer project. Select the table on which you would like to run Clusters.
2. **Results Table Name** – this is a text box for you to enter the name you would like to give to the results/output table. Enter the name in this text box.
3. **Relationship Key Field** – this dropdown requires you to select the key field on which the records are grouped by in your dataset, e.g., Vendor ID, City, etc. If your dataset does not contain such a key field, then you can also select a different field that is unique, e.g., a field with a value that is unique throughout the entire dataset.
4. **Number of Clusters** – this is the number of clusters you are requesting to be created. Based on this number, the data points in your dataset will be grouped into these respective clusters. Because they are grouped into these respective clusters, it also means that they similar to each other based on their characteristics. The 'K' in the name of the algorithm (K-means) represents the number of clusters you want to group your data points into.

Specifying the (right) number of clusters

Choosing/Specifying the right number of clusters could involve some experimentation. The right number of clusters helps ensure that the clustering results are both accurate and useful for your specific application. Choosing the right number is crucial because too few or too many may lead to poor segmentation. To illustrate this, consider the example presented below.

Let's consider a real estate dataset with property prices and square footage to determine the optimal number of clusters.

Property ID	Price $	Sq Footage
1	150,000	1,200
2	180,000	1,500
3	220,000	1,700
4	250,000	2,000
5	600,000	3,500

Property ID	Price $	Sq Footage
6	650,000	3,800
7	700,000	4,000
8	1,200,000	5,500
9	1,300,000	6,000
10	1,500,000	7,000

continued...

Let us now test different values of ‘K’ (number of clusters) to find the best fit.

a. Suppose that **K = 2 (two clusters)**

- Cluster 1: Affordable homes - ~$150k - $250k
- Cluster 2: Luxury homes - ~$600k to $1.5m

   This is useful if we only want a basic price segmentation, e.g., low vs high.

b. Suppose that **K = 3 (three clusters – more granular)**

- Cluster 1: Budget homes - $150k - $250k
- Cluster 2: Mid-range homes - $600k - $700k
- Cluster 3: High-end luxury homes - $1.2m - $1.5m

   This is useful if we want to differentiate mid-range properties.

c. Suppose that **K = 4 (four clusters – more detailed)**

- Cluster 1: Low-end homes - $150k - $220k
- Cluster 2: Mid-range homes - $250k to $600k
- Cluster 3: Premium homes - $650k - $700k
- Cluster 4: Ultra-luxury homes - $1.2m - $1.5m

   This segmentation is great for real estate pricing strategies.

What does this tell us? Based on the clusters created above, it appears that at K = 3 or 4, this probably indicates an optimal number of clusters. If K = 2, we might oversimplify the market and if K = 5 (or above), we might create unnecessary complexity.

Therefore, determining the optimal number of clusters to use can involve some experimentation and testing. The following can help with this determination:

- Understanding your dataset – use commands such as Profile and/or Stratify
- Start off with a higher number of clusters
- Perform the clustering several times specifying a different K value each time
- Reviewing the output results to determine best value for insights and decision-making

5. **Numeric Fields 1 - 4** – this is the numeric field on which you would like to run Clusters. In other words, the data points that you are looking to group into their respective clusters. At a minimum, the procedure requires you to specify at least two numeric fields. Selecting just one numeric field would result in a pop-up message displaying an error and termination of procedure.

Output / Results

Upon completion of the Clusters procedure, an output table (that was specified earlier when you ran Clusters) opens containing the results of the clustering. In this output file, you will see a new field called **kclusters.** The range of values in this field will depend on the number of clusters you had specified in the Clusters dialog. For example, if you had specified 4 clusters, then you should see a range of values from 0 to 3. The remaining fields in the output table are the fields from the original table that you would have selected when your ran Clusters.

You can either sort the resulting file on **kclusters,** or another option would be to summarize on this field and then drill-down on the specific records.

D | ML Functionality: Sentiment Analysis

What is Sentiment Analysis?

Sentiment Analysis, also known as opinion mining, is a natural language processing (NLP) technique used in machine learning (ML) to determine the emotional tone or sentiment expressed in a piece of text. The primary aim is to classify the sentiment as positive, negative, or neutral, though more advanced models can even detect emotions such as anger, joy, or sarcasm. Essentially, sentiment analysis uses NLP and ML technologies to train computer software to analyze and interpret text similarly to humans

This technique is widely used by businesses in various applications, such as analyzing customer feedback, monitoring brand reputation, and conducting market research.

The Arbutus ML Sentiment Analysis App is a procedure that calls a Python procedure (SentimentIntensityAnalyzer) to perform sentiment analysis by identifying the polarity of a given text and returning the results to Analyzer as positive, negative, or neutral.

Example of Sentiment Analysis testing

The example below is a dataset consisting of 10 records and three fields. The third field, Sentiment, is the only field resulting from the sentiment analysis testing via the Python procedure.

Customer Name	Customer Feedback	Sentiment
Roger Lopez	I love this product! It's amazing and works perfectly.	Positive
Carol Bell	Terrible service, I am very disappointed. Was a waste of my time…	Negative
Nathan Moore	The experience was okay, nothing special.	Neutral
Jason Singh	Absolutely fantastic. Highly recommend this product. Great price too	Positive
Lilly Wong	Worst purchase ever. Should not be on shelves! Get rid of it, PLEASE!	Negative
Sam Walters	The service was decent but could be better, I guess.	Neutral
Anita Patel	Sally provided me with exceptional service. She should be promoted!	Positive
Henry Johnson	This is the best thing I've bought all year. I love it!	Positive
Melissa Talbot	Will never ever return to this café. HORRIBLE, HORRIBLE, HORRIBLE	Negative
Ali Abbas	One of the best places I've eaten. Highly recommend. Great service!	Positive

Why is Sentiment Analysis useful?

Performing Sentiment Analysis can be useful to businesses in different ways. For example:

- Analyzing customer reviews helps identify improvement areas and address concerns, enhancing satisfaction.
- Monitoring brand reputation in real-time allows prompt responses to both positive and negative sentiments.
- Understanding customer sentiment aids in product development and aligning offerings with preferences.
- Comparing sentiment with competitors highlights strengths and weaknesses, allowing for strategic decision making.
- Positive sentiment shows campaign success, while negative sentiment indicates needed adjustments.

continued…

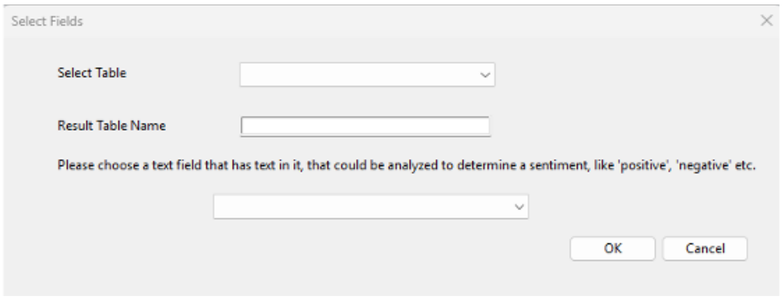
Using the Arbutus ML Sentiment Analysis App

The Arbutus ML Sentiment Analysis App is a procedure that uses Python functionality. Within the Arbutus procedure is a call to a Python procedure/routine (SentimentIntensityAnalyzer) to perform sentiment analysis by identifying the polarity of a given text and returning the results to Analyzer as positive, negative, or neutral.

The ML Sentiment Analysis App can be accessed and run from the Apps Menu: **Apps > AI-ML > ML03 Sentiment Analysis.**

When you run this app, the **Select Fields** dialog is displayed with the following input prompts for you to complete.

1. **Select Table** - this is a dropdown of all the tables in the current Analyzer project. Select the table on which you would like to run Sentiment Analysis.
2. **Results Table Name** – this is a text box for you to enter the name you would like to give to the results/output table. Enter the name in this text box.
3. **Choose a text field** – this is a dropdown of all the character text fields in the current Analyzer project. Select the field on which you would like to be analyzed to determine a sentiment, e.g., positive, negative, neutral



Please choose a text field that has text in it, that could be analyzed to determine a sentiment, like 'positive', 'negative' etc.

Output / Results

Upon completion of the Sentiment Analysis procedure, an output table (that was specified earlier when you ran Sentiment Analysis) opens containing the results of the sentiment analysis. In this output table, you will see two new fields: **Sentiment** and **Sentiment Score**. The remaining fields in the output table are the fields from the original table that you would have selected when your ran Sentiment Analysis.

You can summarize on the **Sentiment** field then drill-down on the specific outcomes. Alternatively, you can apply visualization on the different outcomes.

E | AI Functionality: Smart Query

Use of AI Functionality and ChatGPT in Arbutus

While we provide the Smart Query tool to enable users to interact with their data using natural language, it is important to note that the interpretation and response are handled entirely by the AI engine. Our integrated AI capabilities allow you to ask questions in natural language and receive helpful insights about your data. The clarity, detail, and structure of your questions will influence the quality of the responses, and because AI interprets queries in real time, **the same question may yield different results.** Taking time to learn the basics of prompt engineering—**how to frame questions clearly and provide relevant context**—will help you get more consistent and useful outcomes.

Smart Query is an open-ended, AI-driven interface — it will interpret your input based on the wording, context, and specificity of your question. We encourage you to view AI as a powerful partner in your analysis—one that can accelerate discovery and spark new ideas—**while always applying your own judgment and due diligence to confirm and validate its output.**

Please note that our AI functionality requires internet connectivity and an active ChatGPT account. Otherwise, it is disabled and non-functional by default, to ensures that no data is sent off premise **unless** explicitly decided upon by the customer's organization and users. It will only be enabled once configured with your organization's OpenAI account details. Please take note that the AI Apps will use any information you share via the input prompts or any data from a table that you specify/select, and this information/data is then passed on to OpenAI (via ChatGPT) for subsequent analysis. Therefore, it is imperative that you get confirmation on your organization's policy allowing the use of ChatGPT.

AI-generated content may be incorrect

What is Smart Query?

Smart Query in an AI App that allows the user to ask natural language questions (see 'NLP' in box below) on a dataset that is analyzed, and in return get responses from OpenAI based on interpretation of the questions asked. The AI Smart Query App involves interaction with OpenAI and ChatGPT. Please see the section below for more information.

Natural Language Processing (NLP) is what allows ChatGPT (and other AI tools) to understand what you are saying, i.e. question you are asking, and respond like a human would – using real, natural language.

Running the AI Smart Query App requires the user to specify and enter certain data elements, e.g., the Arbutus table to be analyzed. You will learn more about this in one of the sections below.

Using the Arbutus AI Smart Query App

The Arbutus AI Smart Query App is a procedure that uses Python functionality. Within the Arbutus procedure is a call to a Python procedure/routine **(csv-agent.py)** to perform the requested analysis (based on the query entered by the user) and respond to user request through interaction with ChatGPT.

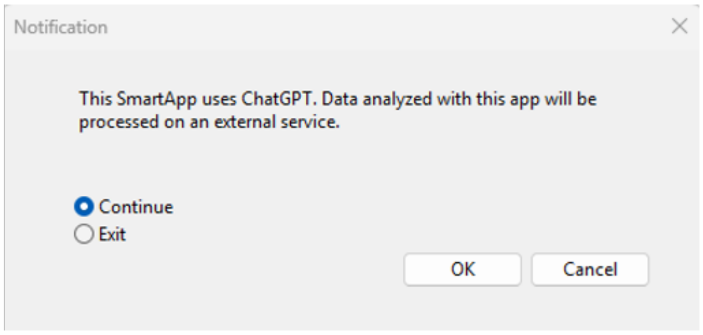
While the AI Smart Query App is launched from an Arbutus procedure, the results are ultimately generated through OpenAI's ChatGPT (via a Python procedure) and then returned to Arbutus for display. Therefore, the results and output are not generated by Arbutus.

continued...

Essentially, the AI Smart Query App uses an Arbutus table, which is selected by the user, and prepares a CSV (delimited) format file for OpenAI to consume. The Python procedure, run from within the Arbutus procedure, passes the CSV format file to OpenAI and initiates an interaction with ChatGPT to perform the requested task that was specified earlier by the user. The results are then passed back to Arbutus.

The AI Smart Query App can be accessed and run from the **Apps** Menu: **Apps > AI-ML > AI01 Smart Query.**

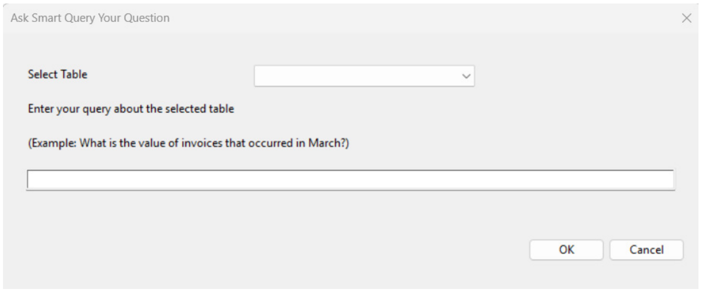
When you run this app, the **Notification** dialog is first displayed to notify you that this App uses ChatGPT, and that data will be processed on an external service (OpenAI). If acceptable, you can click **Continue** followed by **OK.**


Notification

This SmartApp uses ChatGPT. Data analyzed with this app will be processed on an external service.

After clicking **OK** to continue, the **Ask Smart Query Your Question** dialog is displayed.

On this dialog, there are two input prompts for you to complete.


Ask Smart Query Your Question

Select Table

Enter your query about the selected table

(Example: What is the value of invoices that occurred in March?)

1. **Select Table** – this is a dropdown of all the tables in the current Arbutus project. Select the table on which you would like to run Smart Query.
2. **Enter your query about the selected table** – in the text box provided, enter the question that you would like to ask (ChatGPT).

The question that you enter in this text box forms the ChatGPT task.

Framing your questions for ChatGPT interpretation

As mentioned earlier, the AI engine fully manages both the interpretation of your questions and the generation of responses. With our built-in AI features, you can interact with your data using natural language and receive valuable insights. However, the effectiveness of the answers you receive depends on how clearly and specifically your questions are phrased. Because the AI processes each query dynamically, even slight variations in wording can lead to different responses.

To get more reliable and meaningful results, it is helpful to understand the basics of prompt design —the practice of crafting well-structured, clear, and specific questions or instructions that guide the AI toward a desired response. Good prompt design involves using precise language, including relevant details or context, and avoiding ambiguity. This helps the AI better understand your intent and produce responses that are more accurate, consistent, and useful.

Listed below are some examples of how you may want to consider framing your questions for ChatGPT.

> *Note: The examples below are based on fields from a table associated with Healthcare data. Please note that* ***these are just examples and not necessarily standards that you must follow.*** *It is more about providing some form of guidance and assistance for you to think about when constructing your questions. Remember, the effectiveness of the answers you receive depends on how clearly and specifically your questions are phrased. As well, because the AI processes each query dynamically, even slight variations in wording can lead to different responses.*

continued...

#	Suggested Question Form	Comments
a	What is the sum of Charge_Amount for Assigned_Practitioner 'Carlos Gould'. Alternatives: I have a field called Charge_Amount. I would like to find out the sum of this field for a practitioner by the name of Carlos Gould.	In this example, we are specifying the actual field name, e.g., Charge_Amount, as opposed to 'Charge Amount' to avoid any confusion in instances where there is another field name that also has 'Charge in it, e.g., 'Provider's Charge Amount'. As a result, entering the question as 'What is the value of Charge Amount for Carlos Gould' may not give you the correct or desired result. However, being more specific by asking, 'What is the total Charge Amount for Practitioner Carlos Gould' would likely give you the correct result.
b	How many records have a Billing Status of 'Paid'	Not using quotes around 'Paid' should also work. 'Billing Status' is a field that does exist in your dataset.
c	Which insurance provider has the highest Charge_Amount, and what is it?	Using just 'Charge Amount' vs 'Charge_Amount' can avoid confusion over similarly named field names and as such may not return the desired result.
d	I have a date of birth field in my data. I would like to know how many patients were born between 1937 and 1941. Alternatives: How many Patient_date_of_birth between `1937-01-01` AND `1941-12-31`	With reference to the alternative question on the left, we are specifying the actual field name and also specifying the date range in a YYYY-MM-DD format, which should also be acceptable.
e	Which Patient_Name has the largest Number_Days_Admitted. List all the names if more than one	Here we are specifying the field names and also asking to list all the names if more than one was found. Wording this question by providing meaningful and recognizable text closely resembling the actual field names, e.g., 'patient(s)' vs 'Patient_Name' should also yield the correct results.
f	Give me the list of patient names that have the following patient numbers: PN843225, PN271767, PN114579, and PN400181	You should see a list of patient names, e.g., 'Gary Carter, Matthew Johnson, Richard French, and Rita Ramirez'
g	Show me duplicate insurance provider names	You should see a descriptive response, e.g., 'The list of duplicate insurance provider names in the data frame is 'Aetna', 'Humana', 'United Health' '
h	I would like to know how many instances where billing status is 'paid' but where there is no payment date	You should see a descriptive response, e.g., 'The final answer is that there are 7 instances where the billing status is 'paid' but there is no payment date.'
i	Get a list of patient procedure codes ending in '00' and then tell me how many from that list are unique numbers	You should see a descriptive response, e.g. '3 unique numbers'
j	I would like to know the total charge amount (field is Charge_Amount) each for the following Insurance Providers: Aetna, Blue Cross. I would like two separate totals	This question is clearer than saying, 'I would like to know the total charge amount (field is Charge_Amount) for the following Insurance Providers: Aetna, Blue Cross' The above question may just return the total charge amount based on BOTH the insurance providers, Aetna and Blue Cross. Whereas, in the question on the left, we say 'each' in the question and we are clearer when saying 'I would like two separate totals'

continued...

Information and data passed on to OpenAI via the Python Procedure

The following information and data are passed on to ChatGPT for processing:

1. The data from the table that you selected – see step #1 above

Output / Results

Once the results are generated in ChatGPT, they are passed back to Arbutus. Firstly, an Arbutus dialog is displayed showing the result based on the interpretation of the question analyzed by ChatGPT. For example, you may see results like the following displayed in the dialog:

- 19 patients were born between 1937 and 1941.
- The list of duplicate insurance provider names in the data frame is 'Aetna', 'Humana', 'United Health', 'Blue Cross', 'Kaiser Permanente', 'Cigna'
- 96,464.65
- The final answer is that there are 7 instances where the billing status is 'paid' but there is no payment date.

After you click **OK** on this dialog, or after a time-out, the dialog closes and a table called **Response** with a single record, and two fields is opened in the View. The two fields are **Response** and **Question.** In the Response field you will see the same message that was displayed earlier in the dialog and in the **Question** field you will see the question you had entered when you ran the AI Smart Query App earlier.

Note: You can interact with AI in natural language and receive clear, concise text answers. To keep results reliable and easy to use, AI responses are limited to short text only, such as the examples shared above.

IMPORTANT

As the results are generated by OpenAI, it is strongly recommended that users exercise due diligence by reviewing and validating the output within Arbutus to ensure its accuracy and reliability.

F | AI Functionality: Data Categorization

Use of AI Functionality and ChatGPT in Arbutus

While we provide the Data Categorization tool to enable users to interact with their data using natural language, it is important to note that the interpretation and response are handled entirely by the AI engine. Our integrated AI capabilities allow you to ask questions in natural language and receive helpful insights about your data. The clarity, detail, and structure of your questions will influence the quality of the responses, and because AI interprets queries in real time, the **same question may yield different results.** Taking time to learn the basics of prompt engineering—**how to frame questions clearly and provide relevant context**—will help you get more consistent and useful outcomes.

Data Categorization is an open-ended, AI-driven interface — it will interpret your input based on the wording, context, and specificity of your question. We encourage you to view AI as a powerful partner in your analysis—one that can accelerate discovery and spark new ideas—while always **applying your own judgment and due diligence to confirm and validate its output.**

Please note that our AI functionality requires internet connectivity and an active ChatGPT account. Otherwise, it is disabled and non-functional by default, to ensures that no data is sent off premise **unless** explicitly decided upon by the customer's organization and users. It will only be enabled once configured with your organization's OpenAI account details. Please take note that the AI Apps will use any information you share via the input prompts or any data from a table that you specify/select, and this information/data is then passed on to OpenAI (via ChatGPT) for subsequent analysis. Therefore, it is imperative that you get confirmation on your organization's policy allowing the use of ChatGPT.

AI-generated content may be incorrect

What is Data Categorization?

Data categorization, in the context of AI, is the process of organizing and labelling data into predefined categories based on its content, characteristics, or patterns. Data is organized and labelled using a systematic process where each data point is classified into a predefined category based on its characteristics.

The process of categorizing data is performed through a Python procedure called **label-data.py.** This Python procedure is called from within an Arbutus Analyzer procedure, which is the **AI02 Data Categorization** AI-ML App. In addition to data categorization, there also is interaction with ChatGPT and consolidation of responses received from ChatGPT. The interaction with ChatGPT, and the responses received, is based on the task that you specify/enter in the dialog when you run this App. You will learn more about this in the section below.

Using the Arbutus AI Data Categorization App

The Arbutus AI Data Categorization App is an Arbutus procedure that uses Python functionality. Within the Arbutus procedure is a call to a Python procedure/routine **(label-data.py)** to perform data categorization and respond to user request through interaction with ChatGPT.

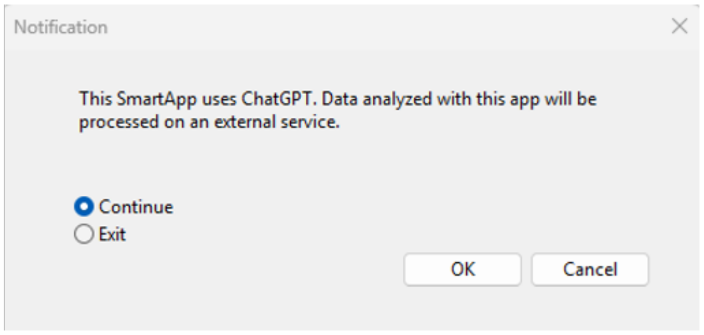
While the AI Data Categorization App is launched from an Arbutus procedure, the results are ultimately generated through OpenAI's ChatGPT (via a Python procedure) and then returned to Arbutus for display. Therefore, the results and output are not generated by Arbutus.

continued...

Essentially, the AI Data Categorization App uses a descriptive field from an Arbutus table, along with two user-defined questions, to initiate an analysis through ChatGPT. This data field and questions are specified by the user. A Python procedure—executed from within the Arbutus environment—passes the selected data field and the questions to OpenAI, triggering an interaction with ChatGPT to carry out the requested task. The results returned by ChatGPT are then passed back to Arbutus and applied accordingly.

The AI Data Categorization App can be accessed and run from the **Apps** Menu: **Apps > AI-ML > AI02 Data Categorization**

When you run this app, the **Notification** dialog is first displayed to notify you that this App uses ChatGPT, and that data will be processed on an external service. If acceptable, you can click **Continue** followed by **OK**.

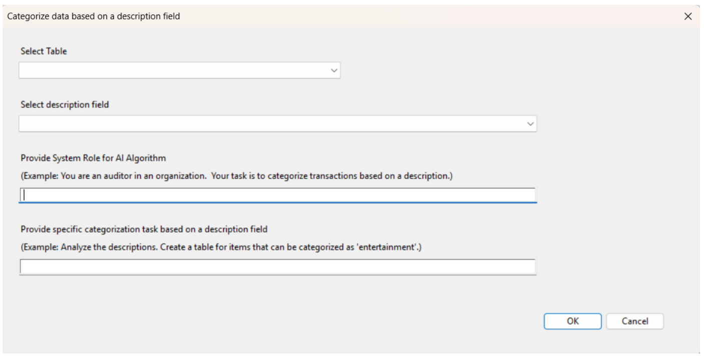

Notification

This SmartApp uses ChatGPT. Data analyzed with this app will be processed on an external service.

[✓ Continue]
Exit

After clicking **OK** to continue, the **Categorize data based on a description field** dialog is displayed.

On this dialog, there are four input prompts for you to complete.


Categorize data based on a description field

Select Table

Select description field

Provide System Role for AI Algorithm
(Example: You are an auditor in an organization. Your task is to categorize transactions based on a description.)

Provide specific categorization task based on a description field
(Example: Analyze the descriptions. Create a table for items that can be categorized as 'entertainment'.)

1. **Select Table** – this is a dropdown of all the tables in the current Analyzer project. Select the table on which you would like to run Data Categorization.
2. **Select description field** – this is a dropdown of all the character fields in the table that you would have just selected. By selecting a description field from here, you are essentially instructing Analyzer, and subsequently the Python procedure (label-data.py), to run the data categorization and ChatGPT analysis based on this field.
3. **Provide System Role for AI Algorithm** – in the text box provided, enter a descriptive text relevant to the activity prompt. It is recommended that you follow a similar structure as the example provided. Here are some other examples:
   - You are an Audit Executive. Your task is to categorize transactions based on Feedback.
   - I have been tasked with categorizing the feedback comments in the selected field.
4. **Provide specific categorization task based on a description field** – in the text box provided, enter a descriptive text relevant to the activity task. It is recommended that you follow a similar structure as the example provided. Here are some other examples:
   - Analyze the Feedbacks. Respond with a table containing items that can be categorized as 'ATM Issues.
   - Analyze the comments in the selected field and create an output containing items categorized as 'Food issues'

The questions that you enter in these two text boxes form the ChatGPT task.

continued...

Information and data that gets passed on to the Python procedure

The following information and data is passed on to the Python procedure for processing:

1. The data from the description field that you selected – see step #2 above
2. The responses you provided in the last-2 input prompts – see step #3 and #4 above

The Python procedure interacts with ChatGPT prepares the responses it receives.

Output / Results

The responses received from ChatGPT is essentially a filtered list of records from the current table (that you had selected) and based on the description field from that same table. For example, if the task you entered/specified was, "Analyze the comments in the selected field and create an output containing items categorized as 'Food issues' ", then you could expect to see a filtered list of all records from the current table (that you had selected) indicating 'Food issues' based on the description field that you had also selected. A new output table is not created.

IMPORTANT

As the results are generated by OpenAI, it is strongly recommended that users exercise due diligence by reviewing and validating the output within Arbutus to ensure its accuracy and reliability.

G | AI Functionality: Arbutus Assistant

Use of AI Functionality and ChatGPT in Arbutus

While we provide the Arbutus Assistant tool to enable users to interact with their data using natural language, it is important to note that the interpretation and response are handled entirely by the AI engine. Our integrated AI capabilities allow you to ask questions in natural language and receive helpful insights about your data. The clarity, detail, and structure of your questions will influence the quality of the responses, and because AI interprets queries in real time, **the same question may yield different results.** Taking time to learn the basics of prompt engineering—**how to frame questions clearly and provide relevant context**—will help you get more consistent and useful outcomes.

Arbutus Assistant is an open-ended, AI-driven interface — it will interpret your input based on the wording, context, and specificity of your question. We encourage you to view AI as a powerful partner in your analysis—one that can accelerate discovery and spark new ideas—while always **applying your own judgment and due diligence to confirm and validate its output.**

Please note that our AI functionality requires internet connectivity and an active ChatGPT account. Otherwise, it is disabled and non-functional by default, to ensures that no data is sent off premise **unless** explicitly decided upon by the customer's organization and users. It will only be enabled once configured with your organization's OpenAI account details. Please take note that the AI Apps will use any information you share via the input prompts or any data from a table that you specify/select, and this information/data is then passed on to OpenAI (via ChatGPT) for subsequent analysis. Therefore, it is imperative that you get confirmation on your organization's policy allowing the use of ChatGPT.

AI-generated content may be incorrect

What is Arbutus Assistant?

Arbutus Assistant in an AI App that allows the user to ask natural language questions (see 'NLP' in box below) to ChatGPT (via OpenAI) and in return get responses from ChatGPT based on interpretation of the questions asked. The AI Arbutus Assistant App involves interaction with OpenAI and ChatGPT. Unlike the AI Smart Query App, the Arbutus Assistant App does not request for an Arbutus table to analyze as part of the ChatGPT request. Please see the section below for more information.

Natural Language Processing (NLP) is what allows ChatGPT (and other AI tools) to understand what you are saying, i.e. question you are asking, and respond like a human would – using real, natural language.

The Arbutus Assistant AI App is an Arbutus procedure that calls a Python procedure called **chatbox-txt.py**, which then does all the processing as requested. This includes the interaction with ChatGPT, and consolidation of responses received from ChatGPT, which are then passed back to Arbutus as the output/results. The interaction with ChatGPT, and the responses received, is based on the task that you specify/enter in the dialog when you run this App. You will learn more about this in the section below.

continued...

Using the Arbutus AI Arbutus Assistant App

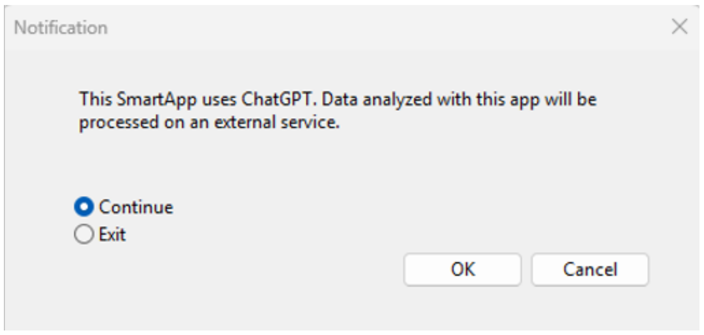
The Arbutus AI Arbutus Assistant App is an Arbutus procedure that uses Python functionality. Within the Arbutus procedure is a call to a Python procedure/routine **(chatbox-txt.py)** to perform the requested analysis by responding to user request through interaction with ChatGPT.

While the AI Arbutus Assistant App is launched from an Arbutus procedure, the results are ultimately generated through OpenAI's ChatGPT (via a Python procedure) and then returned to Arbutus for display. Therefore, the results and output are not generated by Arbutus.

Essentially, the AI Arbutus Assistant App uses a descriptive text, e.g., a question, entered by the user to initiate an analysis through ChatGPT. A Python procedure—executed from within the Arbutus environment—passes the specified question to OpenAI, triggering an interaction with ChatGPT to carry out the requested task. The results returned by ChatGPT are then passed back to Arbutus and displayed in an Arbutus dialog. Although the results are returned to Arbutus, the Arbutus Assistant AI App does allow for a follow-up question that will once again through the same ChatGPT interaction process.

The AI Data Categorization App can be accessed and run from the **Apps** Menu: **Apps > AI-ML > AI03 Arbutus Assistant.**

When you run this app, the **Notification** dialog is first displayed to notify you that this App uses ChatGPT, and that data will be processed on an external service. If acceptable, you can click **Continue** followed by **OK**.

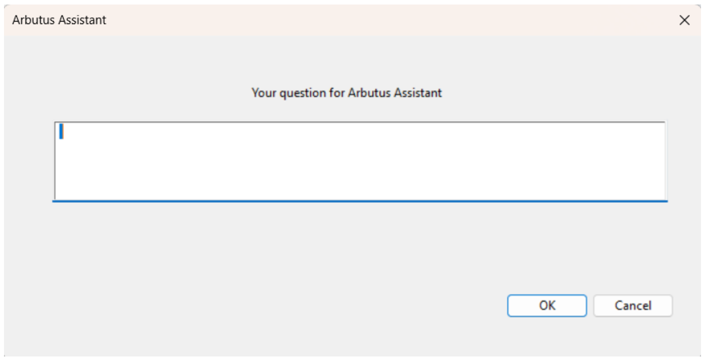

Notification

This SmartApp uses ChatGPT. Data analyzed with this app will be processed on an external service.

[✓ Continue] Exit

After clicking **OK** to continue, the **Arbutus Assistant** dialog is displayed.

On this dialog, there is just one input prompt for you to complete.

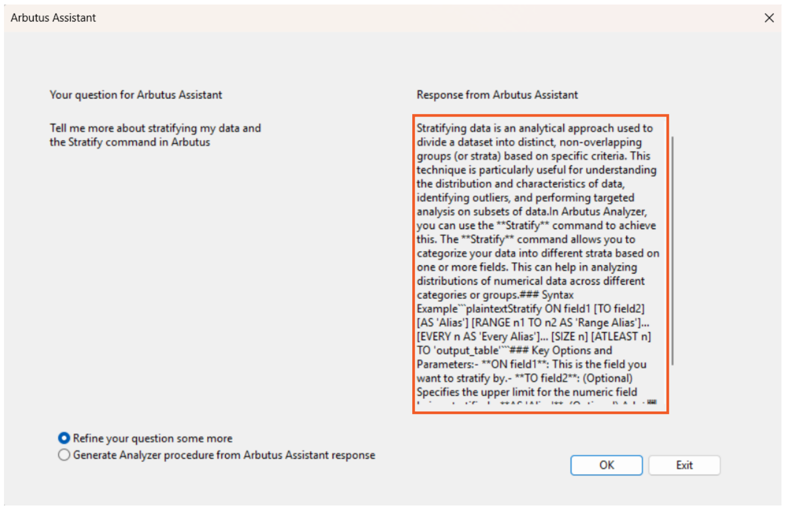


1. **Your question for Arbutus Assistant** – this is a text box for you to enter your question for Arbutus Assistant. Here are some examples:
   - Tell me more about stratifying data in general. What is it about?
   - I am interested in looking for fuzzy duplicates. How do I do this in Arbutus? [Refined question] Are there any Arbutus functions that I should be aware of specific to running fuzzy duplicates?
   - What is the ODBC Administrator?
   - What are the 5 key differences between IDEA and Excel?

continued...

The question that you enter is passed to OpenAI, triggering an interaction with ChatGPT to carry out the requested task. The results returned by ChatGPT are then passed back to Arbutus and displayed in an Arbutus dialog. In the screenshot at right, you can see the results displayed in a text box on the right side of the dialog.

Please note that the following response was generated using AI. While it includes valuable insights, the formatting may not be fully polished and could benefit from further refinement.


Arbutus Assistant
Your question for Arbutus Assistant
Response from Arbutus Assistant

Tell me more about stratifying my data and the Stratify command in Arbutus

~~~~
Stratifying data is an analytical approach used to
divide a dataset into distinct, non-overlapping
groups (or strata) based on specific criteria. This
technique is particularly useful for understanding
the distribution and characteristics of data,
identifying outliers, and performing targeted
analysis on subsets of data.In Arbutus Analyzer,
you can use the **Stratify** command to achieve
this. The **Stratify** command allows you to
categorize your data into different strata based on
one or more fields. This can help in analyzing
distributions of numerical data across different
categories or groups.### Syntax
Example```plaintextStratify ON field1 [TO field2]
[AS 'Alias'] [RANGE n1 TO n2 AS 'Range Alias']...
[EVERY n AS 'Every Alias']... [SIZE n] [ATLEAST n]
TO 'output_table'```### Key Options and
Parameters:- **ON field1**: This is the field you
want to stratify by.- **TO field2**: (Optional)
Specifies the upper limit for the numeric field
~~~~

Refine your question some more
Generate Analyzer procedure from Arbutus Assistant response
OK
Exit

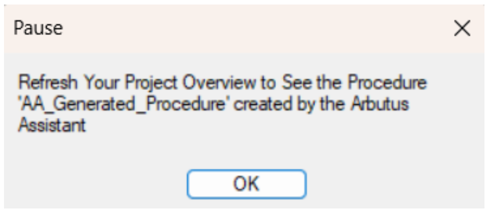
There are two options available on this dialog to continue with the Arbutus Assistant App.

a. **Refine your earlier question some more** – by selecting this option, you have the ability to refine your earlier question. For example, suppose that your earlier question was, "Tell me more about the stratifying my data and the Stratify command in Arbutus", you may want to ask a follow-up question such as, "Can I stratify into 10 buckets?"

   When you select this option, the Arbutus Assistant dialog is displayed. This is the same dialog, with the text box to enter your question, that was initially displayed when you first ran this App. Once again, the question that you enter is passed to OpenAI, triggering an interaction with ChatGPT to carry out the requested task. The results returned by ChatGPT are then passed back to Arbutus and displayed in an Arbutus dialog.

b. **Generate Analyzer procedure from Arbutus Assistant Response** – by selecting this option, you have the ability to generate an Arbutus procedure in the Analyzer that is currently open, containing the response to your question. This merely serves as a repository to store the contents of the response in a more readable and presentable format. It is NOT a procedure that is meant to be executed.

When you select this option, a small pop-up box is displayed with instructions and information on completing this option.



Refresh Your Project Overview to See the Procedure 'AA_Generated_Procedure' created by the Arbutus Assistant

Once you refresh your project (in the Project Overview window), you will notice an Arbutus procedure called **AA_Generated_Procedure** listed in the Project Overview window. You can open this procedure to verify its contents, which you should notice is actually the response to your question but displayed in a more presentable manner.

This procedure will still be listed in the Project Overview even after you close the Arbutus project and re-open it. However, all subsequent selections of this option will result in this same procedure **(AA_Generated_Procedure)** being overwritten. A new procedure is not created each time you run the Arbutus Assistant and select this option.

continued...

Information and data that gets passed on to the Python procedure

With the Arbutus Assistant AI App, no data, e.g., Arbutus project tables, etc., is passed on to the Python procedure for processing. Only the question that you enter at the time when you run the App is passed on.

Output / Results

The output/results received from ChatGPT is essentially a response to the question that was asked to ChatGPT. This response is displayed in an Arbutus dialog. A new output table is not created.

IMPORTANT

As the results are generated by OpenAI, it is strongly recommended that users exercise due diligence by reviewing and validating the output within Arbutus to ensure its accuracy and reliability.

Other questions and/or requests for assistance

There may be times when you need to consult with the technical support team at Arbutus Software. If so, please send an email request to support@ArbutusSoftware.com.

For more information, please refer to the CONTACT US page on our website.

Arbutus Analytics

Arbutus delivers the very best in purpose-built audit analytics technology to meet the exacting demands of today's business environment. Auditors, business analysts, and fraud investigators rely on Arbutus to enhance their testing, analysis and compliance capabilities.

Toll free: 877.333.6336
Phone: 604.437.7873

Sales Inquiries

info@ArbutusSoftware.com

Technical Support

support@ArbutusSoftware.com

ARBUTUS
Powerful Analytics Simplified