DevTreks –social budgeting that improves lives and livelihoods

CTA Algorithm 4, AML Examples

Last Updated: November 23, 2016; First Released: January 09, 2015

Author: Kevin Boyle, President, DevTreks

Version: 2.0.6

A. Algorithm 4 Introduction

The sibling reference, Conservation Technology Assessment (CTA), introduces the background numerical techniques for completing CTAs. This reference introduces examples of CTAs completed using Algorithm 4, Azure Machine Learning (AML) Algorithms.

This algorithm relies on Azure Machine Learning (AML) to run AML algorithms. AML has a variety of algorithms available for analyzing data. The following URLs demonstrate using AML.

https://studio.azureml.net/

http://azure.microsoft.com/en-us/documentation/articles/machine-learning-algorithm-choice/

http://azure.microsoft.com/en-us/documentation/services/machine-learning/

Although the URLs demonstrate using AML directly, this algorithm programmatically runs these types of AML scripts for Data URL datasets and then summarizes the statistical results in the Indicator.MathResult property. The scripts and datasets must be developed and tested in AML by algorithm authors prior to using this algorithm. Datasets and scripts must be cleaned up and tested prior to using them with any CTA algorithm. The same versions of the statistical packages documented in **Section B** must be used.

R and Python algorithms use the same techniques as the examples demonstrated in the CTA 2 and 3 tutorials. The dataset goes in the Data URL and the scripts goes in either the Stock.JointDataURL or M&E.Score.URL property. However, the scripts in Appendix A, Examples 1 and 2 demonstrate that the AML web services require scripts customized for AML conventions –specifically the requirement of producing AML Data Frame results.

AML algorithms, as demonstrated in Appendix A, Example 3, require that paths to “scoring” or “data to score” datasets be added to the Stock.JointDataURL or M&E.Score.URL property along with an optional AML-compatible “training” or “data to train algorithm” datasets that are referenced using the Data URL. If the Data URL is left blank, while the Joint Data property has a good scoring dataset, the algorithm assumes that the statistical model has already been trained by the algorithm author. Statistical models that have already been trained will always run faster than their corresponding non-trained models. Example 3 also demonstrates how to use the M&E calculators upgraded in Version 2.0.6 with this algorithm.

This algorithm does not require that datasets follow the standard explained in the *Resource Stock Calculation* reference. Instead, the datasets should follow AML conventions for csv files (i.e. a header row followed by the raw data). The Data URL dataset must correspond to one and only one indicator (but multiple datasets can be used by following the semicolon delimiter convention). The dataset will be matched with that Indicator based on the dataset column names along with the standard Ix.Qx.ColName convention used in the Indicator.MathExpression.

The current version runs AML algorithms using the following techniques. These web services use batch processing on the cloud site and html Request/Response commands on web servers. For security reasons, AML runs the scripts in a sandbox.

1. **Azure Machine Learning (AML) web service with AML algorithm**: Training and/or scoring data files are sent to an Azure Machine Learning web service that runs the AML algorithm.
2. **Azure Machine Learning (AML) web service with R algorithm**: R Scripts and data files are sent to an Azure Machine Learning web service that runs the algorithm. Documentation for AML includes information about the latest version of R supported.

3. **Azure Machine Learning (AML) web service with Python algorithm**: Python scripts and data files are sent to an Azure Machine Learning web service that runs the algorithm. Documentation for AML includes information about the latest version of Python supported.

Future releases will investigate using more installation techniques as well as supporting additional AML features (i.e. chained scripts and datasets, graphical displays).

The goal of most scripts will be to produce confidence intervals for an Indicator's QTM, QTL, and QTU properties. The algorithm always tries to generate those properties from the last csv row of the results.

A. Subalgorithms and Examples

Appendix A has examples demonstrating each of the following subalgorithms.

subalgorithm1: R scripts: Uses an AML web service to run R algorithms.
subalgorithm2: Python scripts: Uses an AML web service to run Python algorithms.
subalgorithm3: Linear Regression: Uses an AML web service to run an AML Linear Regression algorithm. Additional AML algorithms are not available yet.

Summary

This reference demonstrates how to use AML algorithms to complete CTAs. CTAs may help people to reach decisions that improve their lives and livelihoods.

References (URLs were last accessed September, 2016)

Azure Machine Learning (AML). Microsoft. http://azure.microsoft.com/en-us/documentation/services/machine-learning/

References Note

We try to use references that are open access or that do not charge fees.

Improvements, Errors, and New Features

Please notify DevTreks (devtrekkers@gmail.com) if you find errors in these references. Also please let us know about suggested improvements or recommended new features.

A video tutorial explaining this reference can be found at:

https://www.devtreks.org/commontreks/preview/commons/resourcepack/Technology Assessment 1/1526/none

Appendix A. Algorithm 4 Examples.

This Appendix gives examples demonstrating how to use AML subalgorithms.

Example 1. Algorithm 4. SubAlgorithm1: Probabilistic Statistics: R Project Scripts

URLs:

These datasets are owned by the Carbon Emission Reducers club in the GreenTreks network group. If testing on localhost, switch clubs.

https://www.devtreks.org/greentreks/preview/carbon/input/R OLS 1/2147397537/none

R project Data URL dataset
https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7969/Regress1.csv

R project Stock.JointDataURL or M&E.Score.URL script files
https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7962/R1Cloud.txt

Media URL graphical plots
https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7965/Ex8Plot1.PNG

The first example is for the same linear regression that is used in Example 1h in the CTA 1 reference. The dependent variable in the following regression is *household energy consumption* and the independent variables are *household size* and *household size squared*.

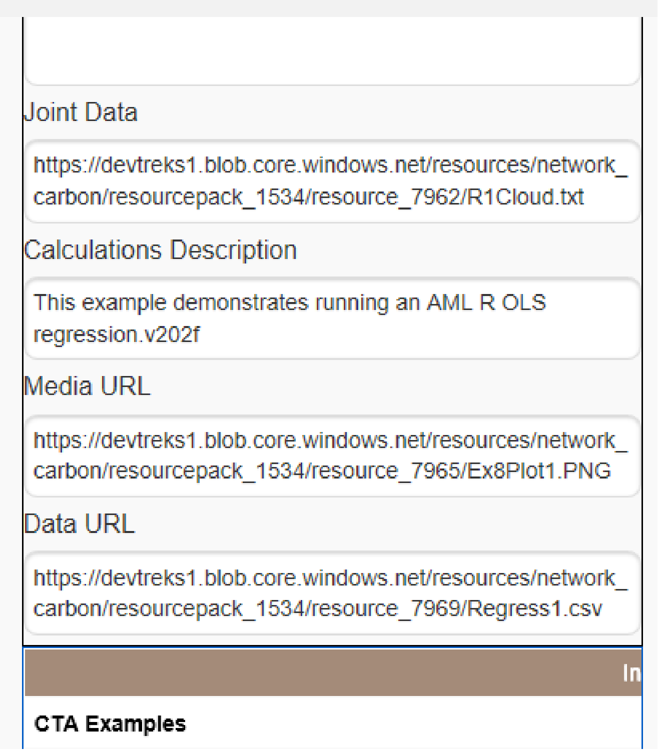
Abstract equation: y = B0 + B1x1 + B2x1^2 + e
Math Expression: I1.Q1.housesize1 + I1.Q2.housesize2

The following script is the AML web service version of this same script. Note that the input data file derives from the Data URL datasets and that an R project dataframe must be produced by the script. Most of the script involves coercing the data results to get the dataframe. The dataframe is saved by the algorithm as a csv file. The csv strings are parsed by the algorithm and displayed in in an Indicator's MathResult, QTM, QTL, and QTU properties. Strings use single quote delimiters.

```
# Map 1-based optional input ports to variables
dataset1 <- maml.mapInputPort(1)
#dataset2 is dataset1 minus the last 3 lines of data
dataset2 <- head(dataset1, n=-3)
#dataset3 is the last 3 lines of data used for ci
dataset3 <- tail(dataset1, n=3)
#statistical model
model <- lm(energyuse ~ housesize1 + housesize2, data=dataset2)
f1 <- c(summary(model))
#title row
t1 <- cbind('estimates', 'std errors', 't values', 'prob..Ts')
#store the coefficients in the summary as a matrix
f2 <- round(as.matrix(f1$coefficients), 4)
#store the r stats
f3 <- cbind('R squared', round(f1$r.squared, 4), 'Adj R squared', round(f1$adj.r.squared, 4))
#store the fstats
f4 <- cbind('F statistic:', round(f1$fstatistic, 4), 'N/A', 'N/A')
ci <- predict(model, dataset3, interval='confidence')
#title row
t5 <- cbind('estimate', 'lower ci  ', 'upper ci', '')
f5 <- cbind(round(ci, 4), 'N/A')
#convert to dataframe for export (tratios, rstats, fstats, cis)
df1 <- data.frame(rbind(t1, f2, f3, f4, t5, f5))
```

# Select data.frame to be sent to the output Dataset port
maml.mapOutputPort('df1');

The following image shows the correct URL conventions to follow using Technique 2 in a Resource Stock Input calculator. The script file URLs end with a ".txt" extension and are stored in Resource elements. The dataset URLs end with a ".csv" extension and are stored in Resource elements. An Internet connection is needed.



Joint Data

https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7962/R1Cloud.txt

Calculations Description

This example demonstrates running an AML R OLS regression.v202f

Media URL

https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7965/Ex8Plot1.PNG

Data URL

https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7969/Regress1.csv

The only properties that are used differently than Example 1h in the CTA 1 reference are as follows:

- **QTM, QTL, and QTU**: All R scripts must include confidence intervals for these properties in the last row of data. That csv-delimited, or space-delimited, string is parsed by the algorithm to generate the Indicator's QTM, QTL, and QTU properties. The scripts above demonstrate one way to generate the interval. Alternatively, any Indicator or Score property can be manually set.

- **Math Type: algorithm4 (AML) and Sub Math Type: subalgorithm1 (R):** Run generic R scripts.
- **Math Expression**: Follows the exact same format as Example 1h in the CTA 1 reference. The Math Expression is coupled with the Data URL dataset column names to determine which Indicator to update with the calculations. At least one Indicator.MathExpression must contain all of the dataset *independent* variable column names. The dependent variable is not included in Math Expressions.

  I1.Q1.housesize1 + I1.Q2.housesize2
- **Math Result**: The R project dataframe output will be saved as a csv TEXT file. If the initial Math Result has a URL to a Resource TEXT file, the csv TEXT file will be stored in the URL. If a URL is not found, the TEXT file will be added to the Math Result property and formatted into columns. The last line of csv text is used to complete QTM, QTL, and QTU. Large mathematical results should always be stored in Math Result URLs.

The following properties show how Score properties are set.

- **Data URL**: The dataset used in Example 1h in the CTA 1 reference has been converted to an R project-compatible csv dataset and referenced using this property. The independent variable column names must match the column names found in one Indicator.MathExpression. Each dataset corresponds to one and only one Indicator (unlike standard datasets). The last 3 rows of data in the dataset are used to score the statistical model that is trained using the remaining rows of data. These 3 rows will be used to conduct a sensitivity analysis of the confidence interval for the estimated dependent variable.
- **Stock.JointDataURL or M&E.Score.URL**: This property stores URLs to one or more R script files that are saved as plain text file with ".txt" extensions in Resource elements. Some scripts may contain characters that translate badly to source code strings (i.e. double quotes around strings). Although error messages use terms like "bad request", any error

should first assume that a script contains characters that cause the error. The cloud script must produce an R dataframe as the result. The web server script must produce a formatted string as the result.

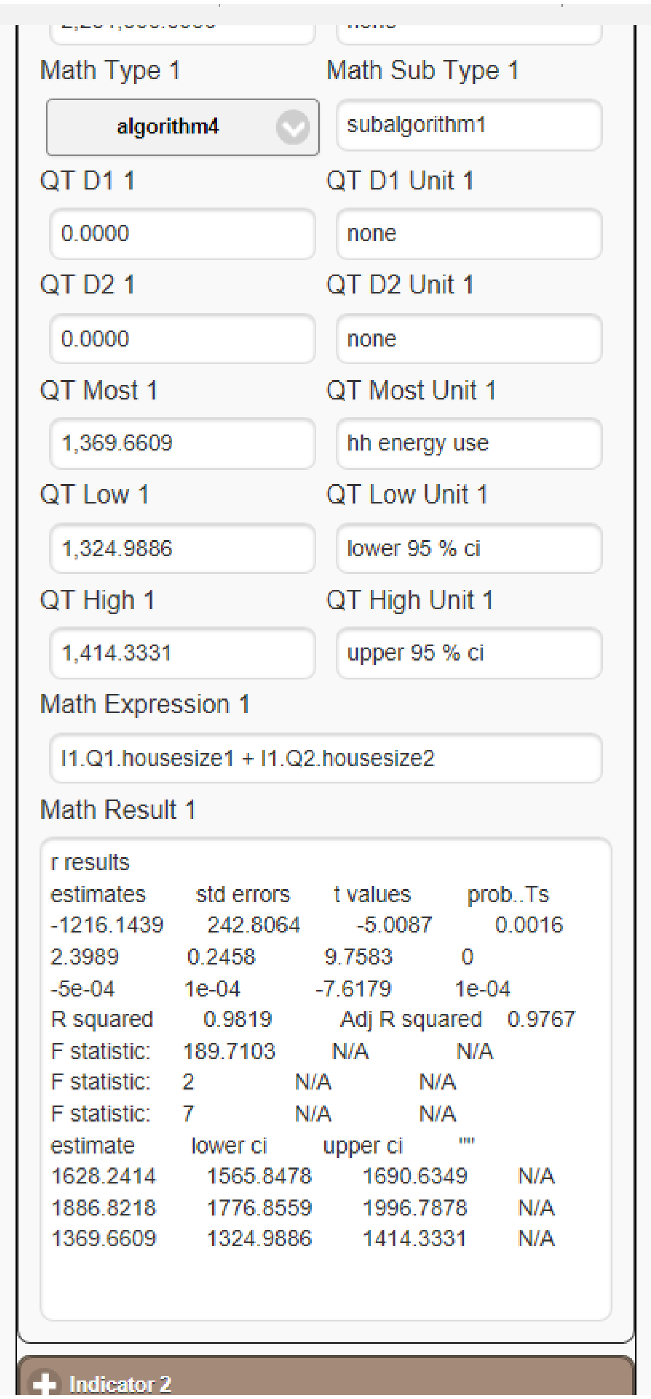
- **Media URL**: These media files include images of graphical plots of the analysis which are generated in R Studio, uploaded to Resource elements, and then referenced here.
- **Remaining Score Properties:** Scores derive from uncertain Indicators and are themselves uncertain. For illustrative purposes, these properties were set to return a similar confidence interval to the sole Indicator being scored. Real scores derive from multiple Indicators and can have probability distributions that are different than the normal distributions used by regression Indicators. The M&E calculators support setting scores using separate datasets. The Stock calculators do not currently support using datasets to set Scores therefore their uncertainty must be analyzed using similar techniques (i.e. algorithm1, subalgorithms 1 to 4).

The calculator uses the following steps:

- **Step 1**. Run an asynchronous loop that simultaneously iterates through each dataset referenced in Data URL. Pass the URL to the algorithm and run the corresponding R Project script for the Indicator corresponding to the iteration loop. The cloud site web service is noticeably slower than the web server executed script, so be patient. Errors with calculations will be added to the Math Result property of each Indicator. To the extent possible, algorithms with multiple datasets run their calculations asynchronously and simultaneously.
- **Step 2**. On cloud sites, the R project script dataframe output will be added to a csv output file. That file will be parsed and displayed in the Indicator.MathResult. Web server sites display the string generated by the R project script.
- **Step 3**. Add the results of the last line of the output file to each Indicator’s QTM, QTL, and QTU properties. Cloud sites parse a csv string and web sites parse a csv or space-delimited string for this purpose.
- **Step 4**. Set the ScoreM property from the result of the Score.MathExpression.

The following results are similar to the results in Example 1h in the CTA 1 reference. The images demonstrates that numbers should be formatted correctly inside the R script because the algorithm does not format any of the statistical results. The Technology Assessment 2 tutorial explains that large mathematical results should always be stored in Math Result URLs.

AML web service Request\Response results



Math Type 1: algorithm4

Math Sub Type 1: subalgorithm1

QT D1 1: 0.0000

QT D1 Unit 1: none

QT D2 1: 0.0000

QT D2 Unit 1: none

QT Most 1: 1,369.6609

QT Most Unit 1: hh energy use

QT Low 1: 1,324.9886

QT Low Unit 1: lower 95 % ci

QT High 1: 1,414.3331

QT High Unit 1: upper 95 % ci

Math Expression 1: I1.Q1.housesize1 + I1.Q2.housesize2

Math Result 1

```
r results
estimates      std errors     t values       prob..Ts
-1216.1439      242.8064       -5.0087        0.0016
2.3989         0.2458         9.7583         0
-5e-04         1e-04          -7.6179        1e-04
R squared       0.9819         Adj R squared   0.9767
F statistic:   189.7103       N/A            N/A
F statistic:   2              N/A            N/A
F statistic:   7              N/A            N/A
estimate       lower ci       upper ci       ""
1628.2414      1565.8478      1690.6349      N/A
1886.8218      1776.8559      1996.7878      N/A
1369.6609      1324.9886      1414.3331      N/A
```

AML web service Batch results

Version 2.0.2 debugged this script by verifying that the batch response from the web service was successful. The actual time needed to receive the batch response is probably too slow, currently, to be practical. Alternatives for running R scripts, such as algorithm2, return results in a reasonable amount of time.

Example 2. Algorithm 4. SubAlgorithm2: Probabilistic Statistics: Python Scripts

URLs:
These datasets are owned by the Natural Resource Stock club in the GreenTreks network group. If testing on localhost, switch clubs.

https://www.devtreks.org/greentreks/preview/carbon/input/Python OLS 1/2147397538/none

Python Data URL dataset
https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7969/Regress1.csv

Python Joint Data URL script files
https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7970/PyOLSCloud1.txt

Media URL graphical plots
https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7966/Ex9Plot1.PNG

The first example is for the same linear regression that is used in Example 1h in the CTA 1 reference. The dependent variable in the following regression is *household energy consumption* and the independent variables are *household size* and *household size squared*.

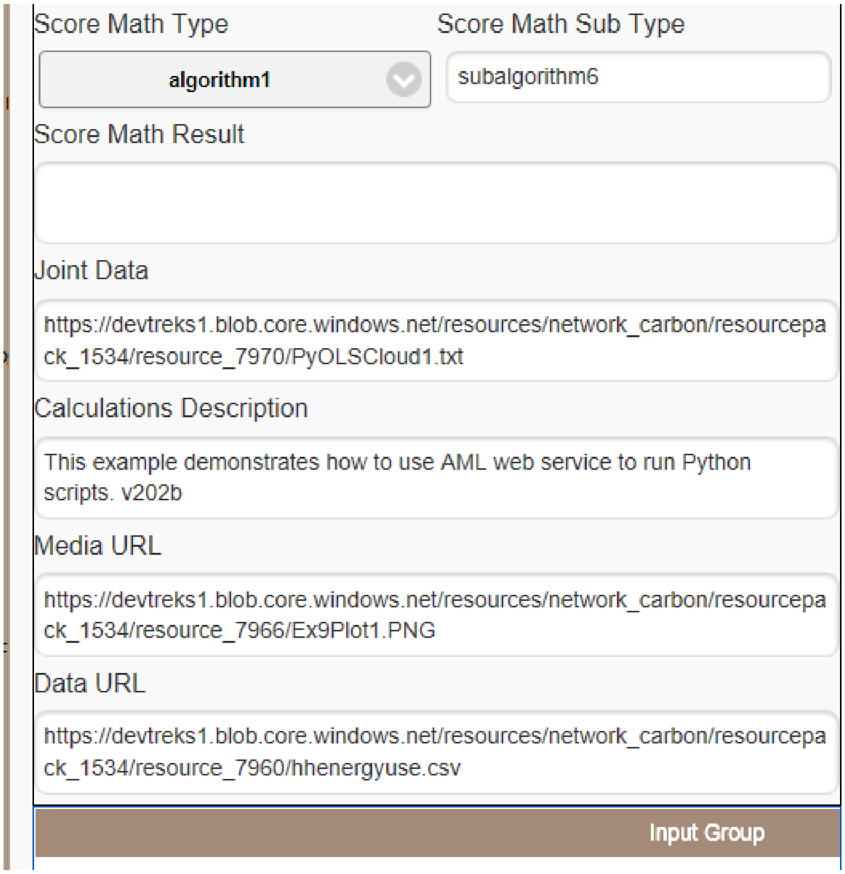
Abstract equation: y = B0 + B1x1 + B2x1^2 + e
Math Expression: I1.Q1.housesize1 + I1.Q2.housesize2

The following script is the AML web service version of a Python OLS algorithms. Note that the first argument passed to the script is a Data URL to a csv dataset. A Python dataframe, known as a panda, must be produced by the script. The dataframe is saved by the algorithm as a csv file. The csv strings are parsed by the algorithm and displayed in an Indicator’s MathResult, QTM, QTL, and QTU properties. Strings use single quote delimiters. This script was debugged using the Anaconda 4.1.1 installation with Python 3.5.2.

```
def azureml_main(dataset1):
    import sys
    import statsmodels.formula.api as smf
    import statsmodels.api as sm
    import pandas as pd

    #dataset2 leaves out the last 3 lines of data used for ci
    dataset2 = dataset1.head(-3)
    #dataset3 is the last 3 lines of data used for ci
    dataset3 = dataset1.tail(3)
    #run statistical model with dataset2
    mod = smf.ols('energyuse ~ housesize1 + housesize2', data=dataset2)
    results = mod.fit()
    #put coefficients, standard error, tvals, pvals into dataframe
    df1 = pd.DataFrame({'A': results.params, 'B': results.bse,'C': results.tvalues, 'D': results.pvalues})

    #get predicted values for dataset3
    ci = results.predict(dataset3)
    #make an exog parameter from a nobs X k array
    colindex = len(dataset3.columns)
    X = dataset3.iloc[:,1:colindex]
    #add the intercept
    X = sm.add_constant(X)
    #get confidence intervals for predicted value of dataset3
    prstd, iv_l, iv_u = wls_prediction_std(results, exog=X, alpha=.05)
    indexcount = len(dataset3.index) - 1

    #put rest of stats into a second df
    df2 = pd.DataFrame(columns=['A', 'B', 'C', 'D'])
    #df2.loc[len(df2.index)] = [round(results.rsquared, 4), round(results.rsquared, 4), 'Adj R squared', round(results.rsquared_adj, 4)]
    df2.loc[len(df2.index)] = ['R squared', round(results.rsquared, 4), 'Adj R squared', round(results.rsquared_adj, 4)]
    df2.loc[len(df2.index)] = ['F statistic:', round(results.fvalue, 4), 'F p val', round(results.f_pvalue, 4)]
    df2.loc[len(df2.index)] = ['predicted ', 'Low 95 CI ', 'High 95 CI', '']
    df2.loc[len(df2.index)] = [round(ci[0], 4), round(iv_l[indexcount - 2], 4), round(iv_u[indexcount - 2], 4), 'N/A']
    df2.loc[len(df2.index)] = [round(ci[1], 4), round(iv_l[indexcount - 1], 4), round(iv_u[indexcount - 1], 4), 'N/A']
    df2.loc[len(df2.index)] = [round(ci[2], 4), round(iv_l[indexcount], 4), round(iv_u[indexcount], 4), 'N/A']
    #py 2.1 workaround
    #return df2;
    #append the df2 to df1
    df3 = df1.append(df2, ignore_index=True)
    #return the dataframe
    return df3;
```

The following image shows the correct URL conventions to follow using Technique 2. The script file URLs end with a ".txt" extension and are stored in Resource elements. The dataset

URLs end with a ".csv" extension and are stored in a Resource. An Internet connection is needed. This script will not run correctly until AML supports Python 4.1.1 (i.e. Score = -.0003).



Score Math Type: algorithm1

Score Math Sub Type: subalgorithm6

Score Math Result:

Joint Data: https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7970/PyOLSCloud1.txt

Calculations Description: This example demonstrates how to use AML web service to run Python scripts. v202b

Media URL: https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7966/Ex9Plot1.PNG

Data URL: https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7960/hhenergyuse.csv

The only properties that are used differently than Example 1h are as follows:

- **QTM, QTL, and QTU**: All Python scripts must include confidence intervals for these properties in the last row of data. That csv-delimited, or space-delimited, string is parsed by the algorithm to generate the Indicator's QTM, QTL, and QTU properties. The scripts above demonstrate one way to generate the interval. Alternatively, any Indicator or Score property can be manually set.
- **Math Type: algorithm4 (AML) and Sub Math Type: subalgorithm2 (Python):** Run generic Python scripts.

- **Math Expression**: Follows the exact same format as Example 1h. The Math Expression is coupled with the Data URL dataset column names to determine which Indicator to update with the calculations. At least one Indicator.MathExpression must contain all of the dataset *independent* variable column names. The dependent variable is not included in Math Expressions.

  I1.Q1.housesize1 + I1.Q2.housesize2
- **Math Result**: The Python dataframe output will be saved as a csv TEXT file. If the initial Math Result has a URL to a Resource TEXT file, the csv TEXT file will be stored in the URL. If a URL is not found, the TEXT file will be added to the Math Result property and formatted into columns. The last line of csv text is used to complete QTM, QTL, and QTU. Large mathematical results should always be stored in Math Result URLs.

The following properties show how Score properties are set.

- **Data URL**: The dataset used in Example 1h has been converted to a Python-compatible csv dataset and referenced using this property. The independent variable column names must match the column names found in one Indicator.MathExpression. Each dataset corresponds to one and only one Indicator (unlike standard datasets). The last 3 rows of data in the dataset are used to score the statistical model that is trained using the remaining rows of data. These 3 rows will be used to conduct a sensitivity analysis of the confidence interval for the predicted dependent variable.
- **Stock.JointData URL or M&E.Score.URL**: This property stores URLs to one or more Python script files that are saved as plain text file with ".txt" extensions in Resource elements. Some scripts may contain characters that translate badly to source code strings (double quotes around strings). Although error messages use terms like "bad request", any error should first assume that a script contains characters that cause the error. The cloud script must produce a Python panda dataframe as the result. The web server script must produce a formatted string as the result.
- **Media URL**: These media files include graphical plots of the analysis which are generated in Python, uploaded to Resource elements, and then referenced here.

- **Remaining Score Properties:** Scores derive from uncertain Indicators and are themselves uncertain. For illustrative purposes, these properties were set to return a similar confidence interval to the sole indicator being scored. Real scores derive from multiple indicators and can have probability distributions that are different than the normal distributions used by regression Indicators. The M&E calculators support setting scores using separate datasets. The Stock calculators do not currently support using datasets to set Scores therefore their uncertainty must be analyzed using similar techniques (i.e. algorithm1, subalgorithms 1 to 4).

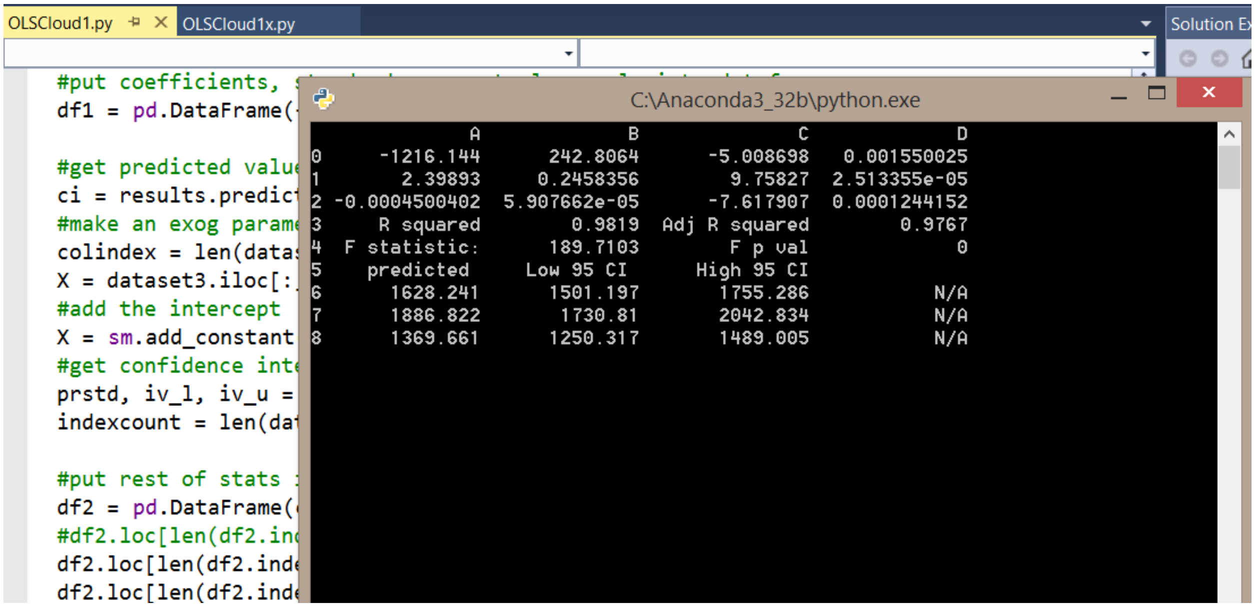
The calculator uses the following steps:

- **Step 1**. Run an asynchronous loop that simultaneously iterates through each dataset in the Data URL TEXT file. Pass the URL to the algorithm and run the corresponding Python script for the indicator corresponding to the iteration loop. The cloud site web service is noticeably slower than the web server executed script, so be patient. Errors with calculations will be added to the Math Result property of each indicator. To the extent possible, algorithms with multiple datasets run their calculations asynchronously and simultaneously.
- **Step 2**. On cloud sites, the Python script dataframe output will be added to a csv output file. That file will be parsed and displayed in the Indicator.MathResult. Web server sites display the string generated by the Python script.
- **Step 3**. Add the results of the last line of the output file to each Indicator's QTM, QTL, and QTU properties. Cloud sites parse a csv string and web sites parse a csv and/or space-delimited string for this purpose.
- **Step 4**. Set the ScoreM property from the result of the Score.MathExpression.

The following results are similar to the results in Example 1h. The images demonstrate that numbers should be formatted correctly inside the Python script because the algorithm does not format any of the statistical results. The algorithm uses the last row to complete an Indicator's

QTM, QTL, and QTU properties (predicted energy use). The Technology Assessment 2 tutorial explains that large mathematical results should always be stored in Math Result URLs.

Python results

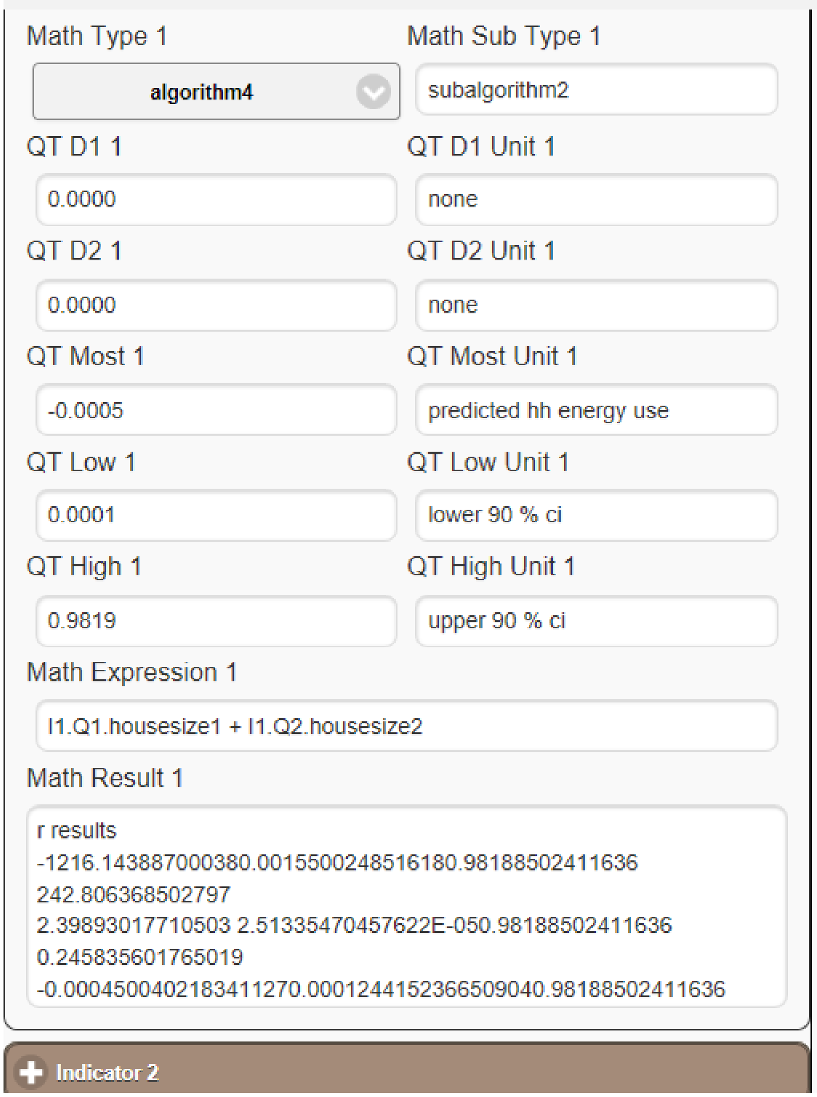


```
#put coefficients, 
df1 = pd.DataFrame(

#get predicted value
ci = results.predict
#make an exog parame
colindex = len(datas
X = dataset3.iloc[:,
#add the intercept
X = sm.add_constant
#get confidence inte
prstd, iv_l, iv_u = 
indexcount = len(dat

#put rest of stats i
df2 = pd.DataFrame(
#df2.loc[len(df2.inc
df2.loc[len(df2.inde
df2.loc[len(df2.inde
```

```
              A            B             C             D
0     -1216.144     242.8064     -5.008698   0.001550025
1       2.39893    0.2458356       9.75827  2.513355e-05
2 -0.0004500402 5.907662e-05     -7.617907  0.0001244152
3     R squared       0.9819  Adj R squared        0.9767
4   F statistic:    189.7103       F p val             0
5     predicted    Low 95 CI    High 95 CI
6      1628.241     1501.197      1755.286           N/A
7      1886.822      1730.81      2042.834           N/A
8      1369.661     1250.317      1489.005           N/A
```

Although the AML web service uses the Python 3.5 option and runs successfully, the following images shows that the service does not return accurate results. The Python version on the web service is believed to be incompatible with this script.

AML web service Request\Response results



Math Type 1	Math Sub Type 1
algorithm4	subalgorithm2
QT D1 1	QT D1 Unit 1
0.0000	none
QT D2 1	QT D2 Unit 1
0.0000	none
QT Most 1	QT Most Unit 1
-0.0005	predicted hh energy use
QT Low 1	QT Low Unit 1
0.0001	lower 90 % ci
QT High 1	QT High Unit 1
0.9819	upper 90 % ci

Math Expression 1

I1.Q1.housesize1 + I1.Q2.housesize2

Math Result 1

```
r results
-1216.143887000380.001550024851618O.98188502411636
242.806368502797
2.39893017710503 2.51335470457622E-050.98188502411636
0.245835601765019
-0.000450040218341127O.00012441523665090.98188502411636
```

Example 3. Algorithm 4. SubAlgorithm3: Probabilistic Statistics: AML Linear Regression

URLs:

These datasets are owned by the Natural Resource Stock club in the GreenTreks network group (if needed, switch clubs).

https://www.devtreks.org/greentreks/preview/carbon/input/AML OLS 1/2147397539/none

AML Data URL training dataset
https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7961/Ex6R.csv

AML Joint Data URL scoring dataset
https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7971/Regress2.csv

Media URL graphical plots
https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7966/Ex9Plot1.PNG

The following URL contains both Resource Stock and M&E calculators that run this example. The URL demonstrates that, outside of the Stock.JointDataURL, and M&E.ScoreURL properties, no changes were needed to run both calculators.

http://localhost:5000/greentreks/preview/carbon/input/Example 4 AML Web Regression/2147409828/none

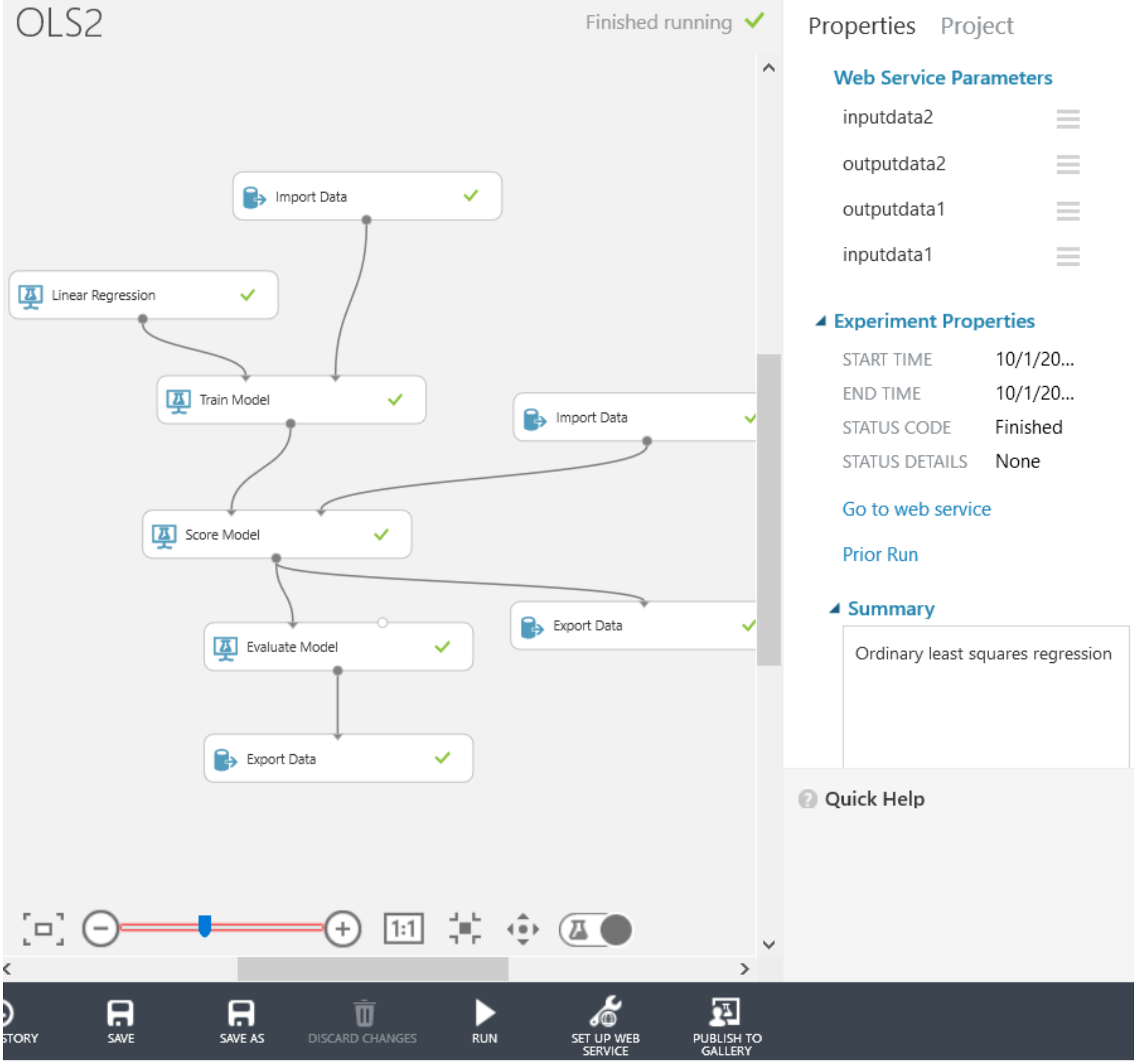
The first example is for the same linear regression that is used in Example 1h in the CTA reference. The dependent variable in the following regression is *household energy consumption* and the independent variables are *household size* and *household size squared*.

Abstract equation: y = B0 + B1x1 + B2x1^2 + e
Math Expression: I1.Q1.housesize1 + I1.Q2.housesize2

The algorithm is passed a Data URL holding the dataset to train the statistical model and a Joint Data URL holding the dataset to score (i.e. with predictions, estimations, or recommendations). The scoring dataset is not used in the regression, but is used to carry out a small sensitivity analysis of the confidence interval for the *predicted* value of the Qx variables (see Example 1h). The final row of the scoring data is used to fill in QTM, QTL, and QTU.

The following URL explains more about the AML Linear Regression algorithm. It also offers an explanation about why the results of this algorithm are not the same as other linear regression algorithms (i.e. glm vs lm regressions in R).

https://msdn.microsoft.com/library/azure/31960a6f-789b-4cf7-88d6-2e1152c0bd1a/
The following image shows what the AML experiment looks like in AML.



Web Service Parameters

inputdata2

outputdata2

outputdata1

inputdata1

Experiment Properties

START TIME	10/1/20...
END TIME	10/1/20...
STATUS CODE	Finished
STATUS DETAILS	None

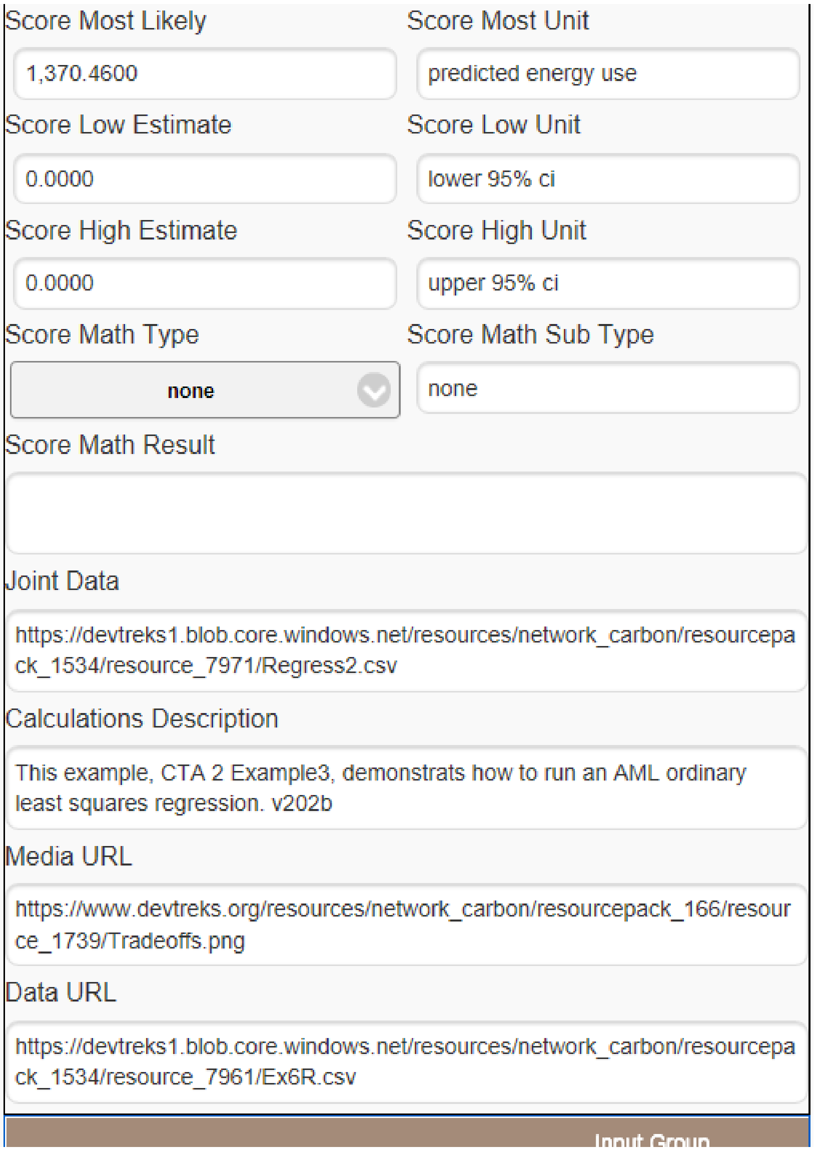
Go to web service

Prior Run

Summary

Ordinary least squares regression

The following image shows the correct URL conventions to follow using this algorithm with a cloud deployment. Both dataset URLs end with a ".csv" extension and are stored in Resource elements. An Internet connection is needed.



Score Most Likely: 1,370.4600

Score Most Unit: predicted energy use

Score Low Estimate: 0.0000

Score Low Unit: lower 95% ci

Score High Estimate: 0.0000

Score High Unit: upper 95% ci

Score Math Type: none

Score Math Sub Type: none

Score Math Result:

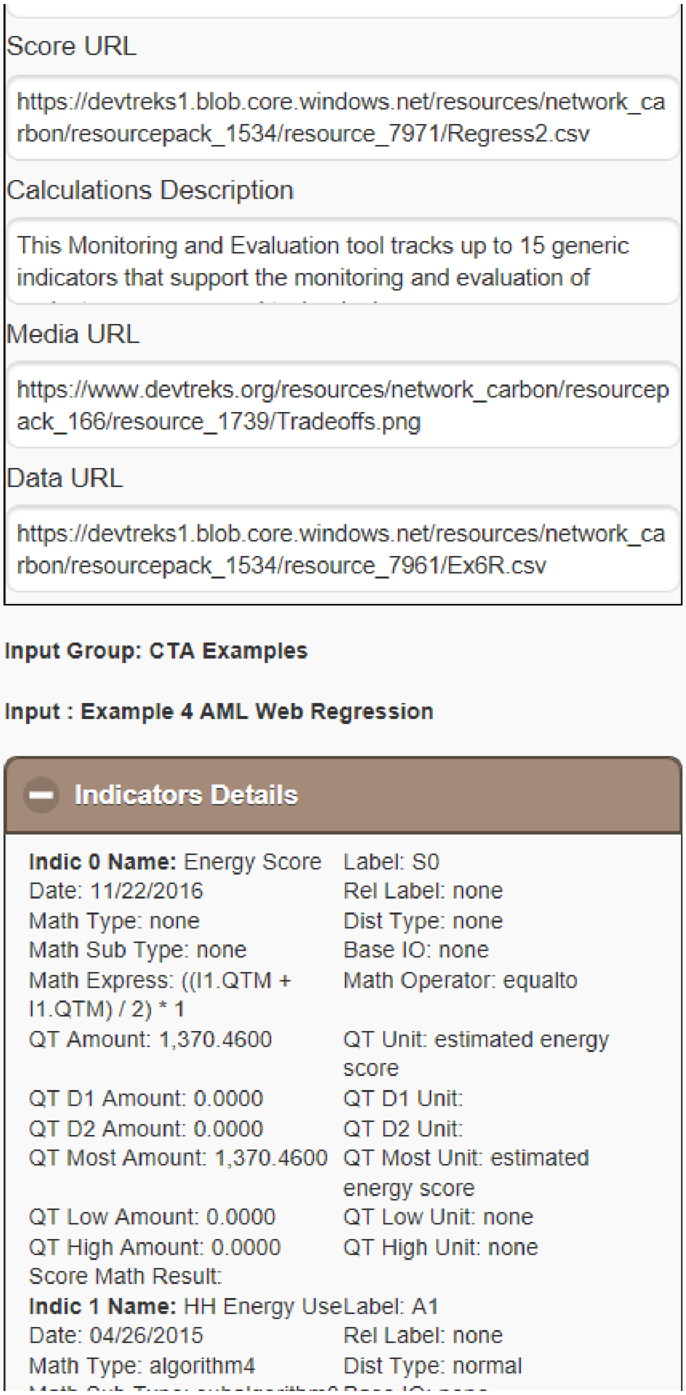
Joint Data: https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7971/Regress2.csv

Calculations Description: This example, CTA 2 Example3, demonstrats how to run an AML ordinary least squares regression. v202b

Media URL: https://www.devtreks.org/resources/network_carbon/resourcepack_166/resource_1739/Tradeoffs.png

Data URL: https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7961/Ex6R.csv

The following image shows the correct file path conventions to follow using Technique 1 with a localhost Intranet deployment in an M&E Input calculator (these files are actually stored on the cloud). The Score.URL property is used to hold the script file.



Score URL

https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7971/Regress2.csv

Calculations Description

This Monitoring and Evaluation tool tracks up to 15 generic indicators that support the monitoring and evaluation of

Media URL

https://www.devtreks.org/resources/network_carbon/resourcepack_166/resource_1739/Tradeoffs.png

Data URL

https://devtreks1.blob.core.windows.net/resources/network_carbon/resourcepack_1534/resource_7961/Ex6R.csv

Input Group: CTA Examples

Input : Example 4 AML Web Regression

Indic 0 Name: Energy Score Label: S0
Date: 11/22/2016 Rel Label: none
Math Type: none Dist Type: none
Math Sub Type: none Base IO: none
Math Express: ((I1.QTM + I1.QTM) / 2) * 1 Math Operator: equalto
QT Amount: 1,370.4600 QT Unit: estimated energy score
QT D1 Amount: 0.0000 QT D1 Unit:
QT D2 Amount: 0.0000 QT D2 Unit:
QT Most Amount: 1,370.4600 QT Most Unit: estimated energy score
QT Low Amount: 0.0000 QT Low Unit: none
QT High Amount: 0.0000 QT High Unit: none
Score Math Result:
Indic 1 Name: HH Energy UseLabel: A1
Date: 04/26/2015 Rel Label: none
Math Type: algorithm4 Dist Type: normal

The only properties that are used differently than Example 1h are as follows:

- **QTM, QTL, and QTU**: All algorithms must generate confidence intervals for these properties in the Math Result string. That csv-delimited string is parsed by the algorithm to generate the Indicator's QTM, QTL, and QTU properties. Alternatively, any Indicator or Score property can be manually set.
- **Math Type: algorithm4 (aml) and Sub Math Type: subalgorithm3 (linear_regression):** Run the AML Linear Regression algorithm.
- **Math Expression**: Follows the exact same format as Example 1h. The Math Expression is coupled with the Data URL dataset column names to determine which Indicator to update with the calculations. At least one Indicator.MathExpression must contain all of the dataset *independent* variable column names. The dependent variable is not included in Math Expressions.

  I1.Q1.housesize1 + I1.Q2.housesize2
- **Math Result**: AML algorithms do not return complete statistical results –they return a small number of scored parameters. These parameters are defined in the following list. The csv string generated by the web service result is used to complete QTM, QTL, and QTU.
  - Mean Absolute Error (MAE): The average of absolute errors (an error is the difference between the predicted value and the actual value). This value is added and subtracted from the predicted QTM to generate QTL and QTU.
  - Root Mean Squared Error (RMSE): The square root of the average of squared errors of predictions made on the test dataset.
  - Relative Absolute Error: The average of absolute errors relative to the absolute difference between actual values and the average of all actual values.
  - Relative Squared Error: The average of squared errors relative to the squared difference between the actual values and the average of all actual values.
  - Coefficient of Determination: Also known as the R squared value, this is a statistical metric indicating how well a model fits the data.
- **Remaining Score Properties:** Scores derive from uncertain Indicators and are themselves uncertain. For illustrative purposes, these properties were set to return a similar confidence interval to the sole indicator being scored. Real scores derive from multiple indicators and

can have probability distributions that are different than the normal distributions used by regression Indicators. The M&E calculators support setting scores using separate datasets. The Stock calculators do not currently support using datasets to set Scores therefore their uncertainty must be analyzed using similar techniques (i.e. algorithm1, subalgorithms 1 to 4).

The following properties show how Score properties are set.

- **Data URL**: The dataset used in Example 1h has been converted to an AML-compatible csv dataset and referenced using this property. The independent variable column names must match the column names found in one Indicator.MathExpression. Each dataset corresponds to one and only one Indicator (unlike standard datasets).
- **Stock.JointData URL or M&E.Score.URL**: This property stores URLs to scoring datasets. The last row of data should correspond to the Qx variables being predicted. The algorithm parses the scored results and fills in QTM, QTL, and QTU with the results.
- **Media URL**: These media files include graphical plots of the analysis which are generated in any statistical package, uploaded to Resource elements, and then referenced here.

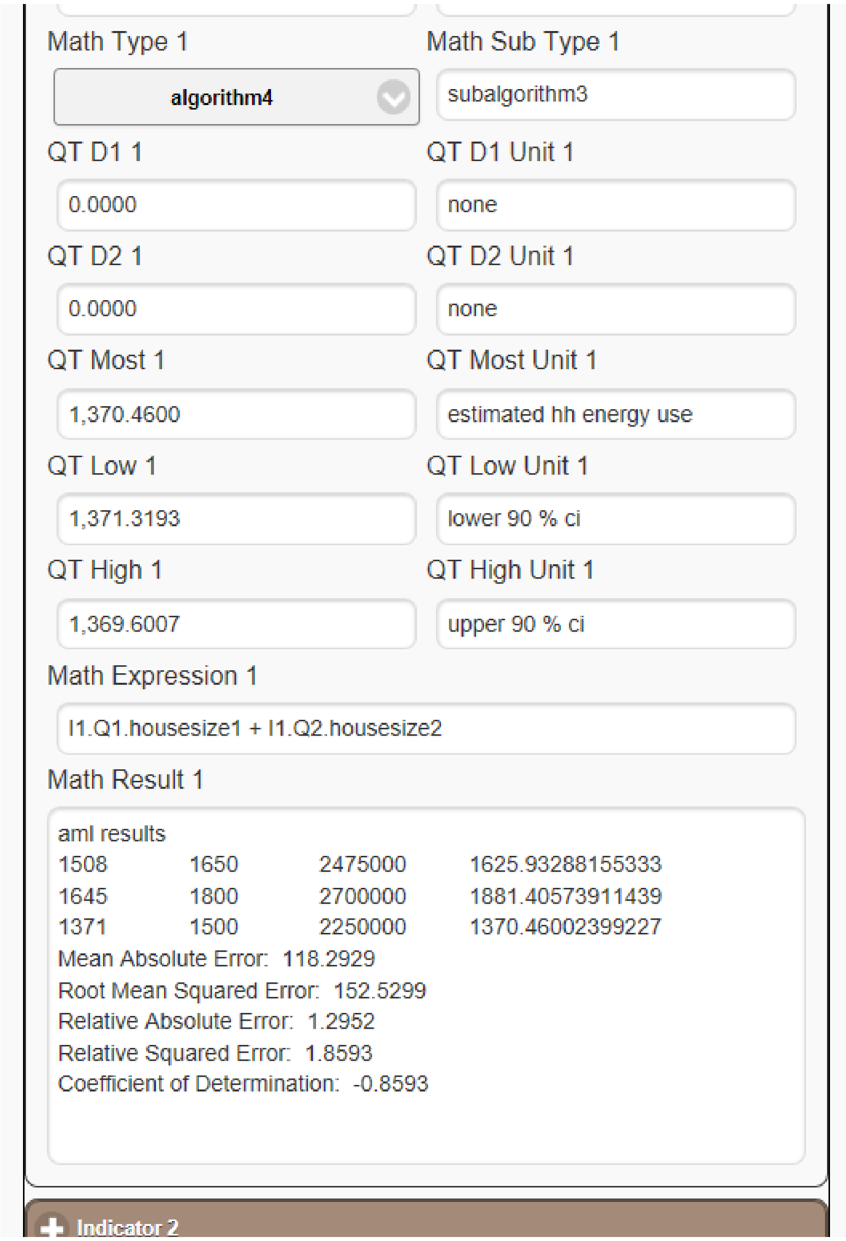
The calculator uses the following steps:

- **Step 1**. Run an asynchronous loop that simultaneously iterates through each dataset in the Data URL TEXT file. Pass the URL to the algorithm and run the corresponding AML algorithm for the indicator corresponding to the iteration loop. The cloud site web service can be slow, so be patient. Errors with calculations will be added to the Math Result property of each indicator. To the extent possible, algorithms with multiple datasets run their calculations asynchronously and simultaneously. The performance of AML algorithms must be considered when deciding which algorithm to use in a CTA.
- **Step 2**. The results of the algorithms will be displayed in the Indicator.MathResult.
- **Step 3**. Add the results of the output file to each Indicator’s QTM, QTL, and QTU properties. Cloud and web sites parse a csv string for this purpose.

- **Step 4**. Set the ScoreM property from the result of the Score.MathExpression.

The following results are similar to the prediction intervals displayed in Example 1h. The algorithm fills out QTM using the last line in the scored dataset (i.e. 1370.4600). The algorithm sets QTL and QTU by subtracting/adding the variable "Mean Absolute Error" from QTM.

AML web service Request\Response results



Math Type 1: algorithm4
Math Sub Type 1: subalgorithm3
QT D1 1: 0.0000
QT D1 Unit 1: none
QT D2 1: 0.0000
QT D2 Unit 1: none
QT Most 1: 1,370.4600
QT Most Unit 1: estimated hh energy use
QT Low 1: 1,371.3193
QT Low Unit 1: lower 90 % ci
QT High 1: 1,369.6007
QT High Unit 1: upper 90 % ci
Math Expression 1: I1.Q1.housesize1 + I1.Q2.housesize2
Math Result 1:

```
aml results
1508        1650        2475000        1625.93288155333
1645        1800        2700000        1881.40573911439
1371        1500        2250000        1370.46002399227
Mean Absolute Error:  118.2929
Root Mean Squared Error:  152.5299
Relative Absolute Error:  1.2952
Relative Squared Error:  1.8593
Coefficient of Determination:  -0.8593
```

AML web service Batch results

Version 2.0.2 debugged this script by verifying that the batch response from the web service was successful. The actual time needed to receive the batch response is probably too slow, currently, to be practical. The techniques that may be developed for algorithm2 and algorithm3 to process long running processes may need to be incorporated here as well.