DataShop Public API

A way to programmatically get and set data in DataShop

The DataShop Team + others
2/6/2015
## Contents

1. API ..................................................................................................................................................... 4
2. Note on format .......................................................................................................................................... 4
   Method Title ......................................................................................................................................... 4
3. What is an ID? ....................................................................................................................................... 4
4. Representational state transfer (REST).............................................................................................. 4
   4.1 HTTP Response Status Codes .................................................................................................... 4
5. Authentication ....................................................................................................................................... 5
   5.1 Access Credentials..................................................................................................................... 5
   5.2 Using your Access Credentials to Sign a Request ................................................................. 5
6. Getting Data ......................................................................................................................................... 10
   6.1 Get Dataset Metadata ............................................................................................................. 10
   6.2 Get Sample Metadata ............................................................................................................. 13
   6.3 Get Transactions ..................................................................................................................... 14
   6.4 Get Student-Step Records ....................................................................................................... 19
7. External Analyses ............................................................................................................................... 24
   7.1 Access and ownership ............................................................................................................. 24
   7.2 Add External Analysis ............................................................................................................. 24
   7.3 Get External Analyses Metadata ............................................................................................. 26
   7.4 Get External Analysis ............................................................................................................... 26
   7.5 Delete External Analysis ......................................................................................................... 27
8. Custom Fields ..................................................................................................................................... 28
   8.1 Access and ownership ............................................................................................................. 28
   8.2 Data Types ................................................................................................................................... 28
   8.3 Add Custom Field .................................................................................................................... 28
   8.4 Get Custom Field Metadata .................................................................................................... 30
   8.5 Set Custom Field ..................................................................................................................... 30
   8.6 Delete Custom Field ................................................................................................................ 31
9. Learning Curve ................................................................................................................................... 32
   9.1 Access ...................................................................................................................................... 32
   9.2 Get Learning Curve .................................................................................................................. 32
Appendix A .................................................................................................................................................. 34
All Possible Result Codes........................................................................................................................................34
Appendix B........................................................................................................................................................36
Revision History........................................................................................................................................36
1 API
This API is a list of methods for programmatically getting and (in the future) modifying DataShop data. The initial target is to provide enough functionality for getting transaction data with various parameters, and adding, modifying, and deleting custom-field data.

As of February 2013, we’ve implemented:

- Authentication
- Get Dataset Metadata
- Get Sample Metadata
- Get Transactions
- Get Student-Step Records
- Add and Get External Analyses

2 Note on format

Method Title

https://url/[?optional_section]
URL to perform the action.

3 What is an ID?
The DataShop API expects you to reference various objects by “ID”, a unique identifier for each dataset, sample, custom field, external analysis, or transaction in the repository. The ID of any of these can be determined by performing a request to list the various items, which lists the IDs in the response. For example, a request for datasets will list the ID of each dataset in the “id” attribute of each dataset element. In the case of a custom fields and external analyses, the ID can also be discovered from the response of a successful request that adds a new custom field or external analysis.

4 Representational state transfer (REST)
The DataShop API design generally follows a "REST" approach to web services. In doing so, we've modeled DataShop as a collection of resources which can be retrieved and manipulated using HTTP. (See the "RESTful Web Services" section of Wikipedia's REST article.)

We do deviate from the REST approach slightly in one way: the DataShop web service ignores the type of HTTP method in each request. That means the service relies on unique URLs with verbs in them (e.g., "get" and "delete") instead of distinguishing requests from similar URLs by HTTP method (e.g., GET or DELETE).

4.1 HTTP Response Status Codes
DataShop Web Services will, in many cases, return an HTTP status code that you can use to determine how to analyze the body of the HTTP message. For example, if you know the HTTP response code was an error such as “401 Unauthorized”, then you know that the content of the body of the message will be XML describing the error.
For a list of web services result codes and the HTTP status codes that appear with them, see Appendix A.

5 Authentication
Authentication is a process for identifying and verifying who is sending a request. To use DataShop web services, you need to identify yourself as the sender of each request. This is accomplished by sending a digital signature that is derived from a pair of public/private access keys.

Note: To use DataShop web services, you must first request access. Visit http://pslcdatashop.web.cmu.edu/WebServicesCredentials to read the User Agreement and request access. You will need to first register with DataShop if you haven’t already.

Note: In addition to this documentation, consult the Java source of the sample DataShop web services client for code examples. See methods signedRequest, encrypt, and httpTimestamp in particular.

5.1 Access Credentials
Access credentials refers to your two access keys. You can retrieve or reset your access keys from DataShop’s Web Services Credentials page.

Access Key ID
Your Access Key ID identifies you as the party responsible for service requests. Include it with each request you send to us.

Secret Access Key
Your Access Key ID has a Secret Access Key associated with it. Use your Secret Access Key to calculate a signature to include in requests to DataShop web services. Your Secret Access Key is a secret, and should be known only by you and DataShop. You should never include your Secret Access Key in your requests to DataShop web services. You should never email your Secret Access Key to anyone. It is important to keep your Secret Access Key confidential to protect your account.

5.2 Using your Access Credentials to Sign a Request
To authenticate to DataShop, you will:

1. Create a request
2. Create a specific type of message signature
3. Send the request and signature to DataShop Web Services

See Figure 1 – Authentication process.

DataShop Web Services will then:

1. Retrieve your Secret Access Key
2. Create the same type of signature
3. Compare the two signatures

If the two signatures match, the request is considered authenticated; if they fail to match, then the request fails authentication.
Step 1: Create a request

In this step, you create an HTTP request in your program which has a standard request line, a number of request headers, and an optional message body (some DataShop Web Services requests require a body while others are only a URL).

To authenticate, your request must contain the following request headers:

- `date`
- `authorization`

Set “date” according to the `timestamp` format (see Table 1 - Contents of the string to sign).

You will set the value of “authorization” in Step 3, described below.

Step 2: Create a message signature

Each request you send must include an HMAC-SHA signature calculated with your Secret Access Key. **HMAC-SHA** is an industry-standard message authentication procedure that uses the **SHA-1** cryptographic hash function in combination with a secret key to create a message signature. DataShop uses it to verify the authenticity of your request.

The signature must be URL-encoded in UTF8. This signature will be included in the “authorization” header as described in Step 3.
The string you use to calculate the signature (the \textit{string to sign}) is composed as follows:

- \textit{method}
- \textit{contentMD5}
- \textit{contentType}
- \textit{timestamp}
- \textit{path}

Each line in the string to sign is separated by a line break. See Table 1 - Contents of the string to sign for a description of each line.

Note: See the Java source of the \texttt{sample DataShop web services client} for a code example.

<table>
<thead>
<tr>
<th>part of the string</th>
<th>description</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>method</td>
<td>the HTTP method used—GET, PUT, POST, or DELETE</td>
<td>GET</td>
</tr>
<tr>
<td>contentMD5</td>
<td>an MD5 hash of the message content if PUT or POST, an empty string if GET or DELETE</td>
<td></td>
</tr>
<tr>
<td>contentType</td>
<td>MIME type of content if PUT or POST, an empty string if GET or DELETE</td>
<td>text/xml</td>
</tr>
<tr>
<td>timestamp</td>
<td>an HTTP date string (see \texttt{httpTimestamp} method below)</td>
<td>Tue, 20 Oct 2009 15:22:52 GMT</td>
</tr>
<tr>
<td>path</td>
<td>the portion of the request URL that is after “services” and before the first question mark (“?”).</td>
<td>/datasets/1/samples/3</td>
</tr>
</tbody>
</table>

```java
/**
 * Format for HTTP date strings.
 * @return format for HTTP date strings
 */
private String httpTimestamp() {
    if (httpDateFmt == null) {
        httpDateFmt = new SimpleDateFormat("EEE, dd MMM yyyy HH:mm:ss zzz");
        httpDateFmt.setTimeZone(TimeZone.getTimeZone("GMT"));
    }
    return httpDateFmt.format(new Date());
}
```

Finally, you must \texttt{URL-encode} the signature you’ve created in UTF-8 character encoding. This is the application/x-www-form-urlencoded MIME format. Your programming language may have a utility method that will perform this encoding for you. For reference, see the Java class \texttt{java.net.URLEncoder}, which you can use to perform UTF-8 URL-encoding in Java.

\textbf{Step 3: Send the request and signature to DataShop Web Services}

After calculating the signature and URL-encoding it, you can now attach it to the request.
Set the “authorization” header as follows, where publicApiToken is your public token and signature is the HMAC-SHA signature you create in Step 2.:

“DATASHOP ” + publicApiToken + “:” + signature

For example:

DATASHOP OKLFETPSJZJJFSGR6D8E:wzXDj0%B3Ey3iosiwcgH%2fsYktDZPM%3D%0D%0A

Note the space after “DATASHOP”.

Send the complete request to DataShop Web Services.

5.2.1 Example

You want to retrieve a description of the first sample (“All Data”) for the dataset with id “1”.

The request line looks like this:

GET /services/datasets/1/samples/1 HTTP/1.1

The request body is empty.

You create a “date” header:

date: Tue, 20 Oct 2009 16:59:47 GMT

Then create the string to sign:

method: GET
ccontentMD5: <empty>
ccontentType: <empty>
ttimestamp: Tue, 20 Oct 2009 16:59:47 GMT
path: /datasets/1/samples/1

or

GET

Tue, 20 Oct 2009 16:59:47 GMT
/datasets/1/samples/1

Create the HMAC-SHA signature from the string to sign and URL-encode it in UTF-8:

wzXDj0%B3Ey3iosiwcgH%2fsYktDZPM%3D%0D%0A

And with that, form the “authorization” header:

authorization: DATASHOP OKLFETPSJZJJFSGR6D8E:wzXDj0%B3Ey3iosiwcgH%2fsYktDZPM%3D%0D%0A

The full request will then look like this:
GET /services/datasets/1/samples/1 HTTP/1.1
date: Tue, 20 Oct 2009 16:59:47 GMT
authorization: DATASHOP OKLFETPSJZJ3FSGR6D8E:wzXDJ0%2BEY31osiwcgH%2FsYktDZPM%3D%0D%0A
accept: text/xml
user-agent: Java/1.6.0_13
host: https://pslcdatashop.web.cmu.edu
connection: keep-alive
6 Getting Data

6.1 Get Dataset Metadata

http://pslcdatashop.web.cmu.edu/services/datasets/[?id]
Get list of datasets matching the parameters, or information about a single dataset (if an id is specified).

6.1.1 Request Parameters

access
all, viewable or editable. Default is viewable. All means return all the datasets, those that are public, view, edit, admin, or private (see descriptions below). Viewable means return datasets for projects on which you have “view”, “edit”, or “admin” privileges, in addition to public datasets. Editable means return only datasets for projects on which you have the “edit” or “admin” privilege.

Note: If you request a single dataset and specify a value for the “access” parameter, DataShop will respect the access parameter. This will result in an empty set in the case that the dataset requested does not match the access parameter specified (e.g., you specified “access=editable” but the dataset is only viewable).

verbose
ture or false. Default is false. False returns the simpler, less verbose description for each dataset (see directly below) while true returns the more verbose description for each dataset (see further below). A verbose response is only possible for datasets you can view or edit (i.e., not private datasets).

6.1.2 Response Fields

access is your level of access to the dataset. DataShop models each user’s dataset access as either “public”, “view”, “edit”, “admin”, or “private”, which is the level of access they have to the dataset’s containing project.

- public: you and any other registered DataShop user can view the data and download associated files
- view: you can view the data and download associated files.
- edit: in addition to the above, you can create samples, add or modify papers and files, add kc sets, and add kc models
- admin: in addition to the above, you can edit dataset and project metadata, add or modify datasets in the project, and manage project access.
- private: you cannot view the data or edit anything about the dataset

For a full description of the access levels, see http://pslcdatashop.org/help?page=administration

number_of_samples is the total number of samples including those that are accessible and those that are not shared.
**number_of_accessible_samples** is the number of samples that you can access, ignoring private samples owned by others.

**number_of_unique_steps** is the number of unique steps, where uniqueness is defined as a step within a specific problem hierarchy (the curriculum location where the problem appears). The same step attempted by two students equals only one unique step.

**Note:** The order of datasets in the response XML is indeterminate.

### 6.1.3 Example request:
GET https://pslcdata.shop.web.cmu.edu/services/datasets/31

### 6.1.4 Example response on success:
```xml
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message result_code="0" result_message="Success.">
<dataset id="31">
    <name>Pittsburgh Science of Learning Center Stoichiometry Study 1</name>
    <project>Stoichiometry Studies</project>
    <domain>Science</domain>
    <learnlab>Chemistry</learnlab>
    <pi>bleber</pi>
    <start_date>2005-09-02</start_date>
    <end_date>2006-06-07</end_date>
    <status>complete</status>
    <access>edit</access>
    <public>yes</public>
    <released>yes</released>
    <number_of_students>34</number_of_students>
    <number_of_unique_steps>16453</number_of_unique_steps>
    <number_of_steps>124882</number_of_steps>
    <number_of_transactions>245093</number_of_transactions>
    <number_of_samples>17</number_of_samples>
    <number_of_accessible_samples>3</number_of_accessible_samples>
    <number_of_kc_models>4</number_of_kc_models>
</dataset>
</pslc_datashop_message>
```

### 6.1.5 Example response on error:
```xml
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message result_code="-1" result_message="Error. Dataset 31 is not valid." />
```

### 6.1.6 Example request (verbose)
GET https://pslcdata.shop.web.cmu.edu/services/datasets/31?verbose=true

### 6.1.7 Example verbose XML response (more metadata):
```xml
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message result_code="0" result_message="Success.">
<dataset id="31">
    <name>Pittsburgh Science of Learning Center Stoichiometry Study 1</name>
</dataset>
</pslc_datashop_message>
```
<project>Stoichiometry Studies</project>  
<domain>Science</domain>  
<learnlab>Chemistry</learnlab>  
<pi>bleber</pi>  
<start_date>2005-09-02</start_date>  
<end_date>2006-06-07</end_date>  
<status>complete</status>  
<access>edit</access>  
<public>yes</public>  
<released>yes</released>  

<curriculum>geometry 2005</curriculum>  
<tutor></tutor>  
<description></description>  
<has_study_data>yes</has_study_data>  
<hypothesis>Lorem ipsum</hypothesis>  
<school>Wilkinsburg High School</school>  
<additional_notes>School demographics for 2005-6 unless noted otherwise.  
Converter Version 4.15  
Loaded to production on 4/15/08 by Kyle.  
</additional_notes>  
<number_of_students>34</number_of_students>  
<number_of_unique_steps>16453</number_of_unique_steps>  
<number_of_steps>124882</number_of_steps>  
<number_of_transactions>245093</number_of_transactions>  
<number_of_samples>17</number_of_samples>  
<number_of_accessible_samples>3</number_of_accessible_samples>  
<number_of_kc_models>4</number_of_kc_models>  

<kappa id="25">  
<name>Automatic-Model</name>  
<number_of_kcs>456</number_of_kcs>  
<observations_with_kcs>470117</observations_with_kcs>  
<logistic_regression_model_status>unable to run</logistic_regression_model_status>  
<cross_validation_status>unable to run</cross_validation_status>  
</kc_model>  

<kappa id="32">  
<name>Single-KC</name>  
<number_of_kcs>1</number_of_kcs>  
<observations_with_kcs>126057</observations_with_kcs>  
<number_of_parameters>10</number_of_parameters>  
<logistic_regression_model_status>complete</logistic_regression_model_status>  
<aic>142117.54</aic>  
<bic>142468.24</bic>  
<log_likelihood>-2759.55</log_likelihood>  
<cross_validation_status>complete</cross_validation_status>  
<cross_validation_rmse>0.3457</cross_validation_rmse>  
<cross_validation_number_of_observations>126056</cross_validation_number_of_observations>  
<cross_validation_number_of_parameters>10</cross_validation_number_of_parameters>  
</kc_model>  

<kappa id="14">  
<name>Unique-step</name>  
<number_of_kcs>9157</number_of_kcs>  
<observations_with_kcs>0</observations_with_kcs>
6.2  Get Sample Metadata

https://pslcdatashop.web.cmu.edu/services/datasets/[id]/samples/[?id]
Get the list and descriptions of samples matching the parameters.

6.2.1 Request Parameters

access
viewable or editable. Default is viewable. Viewable means return only samples you can view. Editable means return only samples you own and can therefore edit.

Note: If you request a single sample and specify a value for the “access” parameter, DataShop will respect the access parameter. This will result in an empty set in the case that the sample requested does not match the access parameter specified (e.g., you specified “access=editable” but the sample is only viewable).

verbose
true or false. Default is false. False returns the simpler, less verbose description for each sample (see directly below) while true returns the more verbose description for each sample (see further below), including a description of each filter in each sample.

6.2.2 Example request:
GET https://pslcdatashop.web.cmu.edu/services/datasets/422/samples/37

6.2.3 Example XML response for a sample query:
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message result_code="0" result_message="Success.">
  <sample id="37">
    <name>CWCTC-GeoArea06-07</name>
    <description>Just the CWCTC students on the area units (GEO-PA)</description>
    <owner>aleven</owner>
    <number_of_transactions>350384</number_of_transactions>
  </sample>
</pslc_datashop_message>

6.2.4 Example verbose XML response for a sample query:
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message result_code="0" result_message="Success.">
  <sample id="37">
    <name>CWCTC-GeoArea06-07</name>
    <description>Just the CWCTC students on the area units (GEO-PA)</description>
  </sample>
</pslc_datashop_message>
6.2.5 Example response on error:
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message
    result_code="-4"
    result_message="Error. Sample 37 is not valid for dataset 422." />

6.3 Get Transactions

https://pslcdatareshop.web.cmu.edu/services/datasets/[id]/[?samples/id]/transactions
Returns zero or more transaction records in tab-delimited form, optionally zipped. If a sample is not
specified, then the “All Data” sample is returned. Note that the specified dataset must be released for
you to get its transactions.

See “Interacting with Cached Data” note below.

6.3.1 Request Parameters

zip
ture or false. Default is false. If true, transaction data is zipped.

cols
Optional list of column names, comma-delimited. Default is to include all columns. See Table 2 -
Transaction columns below for the list of column values to choose from. To specify whether or not
to include custom-field columns, use the cfs parameter.

cfs
all, none, or a comma-delimited list of custom field IDs. Default is none. Optionally specify whether
to include all custom fields, no custom fields, or a subset of custom fields. A custom field is a user-
created column of data associated with transactions. See section 8 of this API for a description of
how to create and modify custom fields, as well as how to retrieve a description of custom fields for
a dataset.

headers
ture or false. Default is true. Optionally specify whether a header row should be included in the
output. If false, a header row is omitted.

limit
Maximum number of transactions to retrieve. Default is 100, maximum is 5,000.
offset

From the beginning of the data, the number of transaction rows to skip. Default is 0. For example, an offset of 0 would return rows starting with the 1st row, while an offset of 100 would return rows starting with the 101st row. Loop through transactions in batches by combining offset and limit. For example, to get rows in batches of 100, request offset=0&limit=100 in the first iteration of the loop, which will return rows 1-100. Then in the next iteration of the loop, request offset=100&limit=100, which will return rows 101-200.

6.3.2 Example request (tab-delimited format, subset of columns):
GET
https://pslcdatashop.web.cmu.edu/services/datasets/114/samples/21/transactions?cols=row,anon_student_id,session_id,time,duration,student_response_type,tutor_response_type,problem_name,step_name,attempt_at_step,outcome,selection,input,feedback,kcs

6.3.3 Example response:
See Table 4 - Example tab-delimited transaction data.

6.3.4 Example response on error:
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message
     result_code="-7"
     result_message="Error. Invalid column: feedbacks." />

Interacting with Cached Data

DataShop maintains a cached export file for each sample, which is used to provide data when you request it. Due to changes made in the web application or more data being logged, the cached file can fall out-of-date with what is available in the DataShop database. When this happens, a request for the sample will return an error stating that the cached file is not yet available. DataShop will then start caching the sample (if it hasn’t already). You should try your request again after some time has passed. The amount of time needed for caching varies between a few minutes and a few hours, depending on the size of the sample.
<table>
<thead>
<tr>
<th>Column name in request</th>
<th>Column(s) contained in response</th>
<th>Data type</th>
<th>Notes on value</th>
</tr>
</thead>
<tbody>
<tr>
<td>row</td>
<td>Row</td>
<td>numeric</td>
<td></td>
</tr>
<tr>
<td>tx_id</td>
<td>Transaction Id</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>anon_student_id</td>
<td>Anon Student Id</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>session_id</td>
<td>Session Id</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>time</td>
<td>Time</td>
<td>date</td>
<td>yyyy-MM-dd HH:mm:ss</td>
</tr>
<tr>
<td>time_zone</td>
<td>Time Zone</td>
<td>string</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>duration</td>
<td>Duration (sec)</td>
<td>numeric</td>
<td>Null values are represented with a dot (“.”)</td>
</tr>
<tr>
<td>student_response_type</td>
<td>Student Response Type</td>
<td>string</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>student_response_subtype</td>
<td>Student Response Subtype</td>
<td>string</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>tutor_response_type</td>
<td>Tutor Response Type</td>
<td>string</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>tutor_response_subtype</td>
<td>Tutor Response Subtype</td>
<td>string</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>problem_hierarchy</td>
<td>Problem Hierarchy</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>problem_name</td>
<td>Problem Name</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>problem_view</td>
<td>Problem View</td>
<td>numeric</td>
<td></td>
</tr>
<tr>
<td>problem_start_time</td>
<td>Problem Start Time</td>
<td>date</td>
<td>yyyy-MM-dd HH:mm:ss</td>
</tr>
<tr>
<td>step_name</td>
<td>Step Name</td>
<td>string</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>attempt_at_step</td>
<td>Attempt At Step</td>
<td>numeric</td>
<td>Possibly empty if step name not provided in data.</td>
</tr>
<tr>
<td>outcome</td>
<td>Outcome</td>
<td>string</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>selection</td>
<td>Selection</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>action</td>
<td>Action</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>input</td>
<td>Input</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>feedback</td>
<td>Feedback Text, Feedback Classification</td>
<td>string, string</td>
<td>Both possibly empty</td>
</tr>
<tr>
<td>help_level</td>
<td>Help Level</td>
<td>string</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>total_num_hints</td>
<td>Total # Hints</td>
<td>numeric</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>condition</td>
<td>Condition Name, Condition Type</td>
<td>string, string</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>kcs</td>
<td>KC (KC Model Name), KC Category (KC Model Name)</td>
<td>string</td>
<td>Possibly empty</td>
</tr>
</tbody>
</table>

**Note:** For more documentation on these columns and the semantics of the format, see the [Import File Verification Tool documentation](#) and [Export By Transaction](#) as documented in the DataShop web application.
Table 3 – Custom field columns included when the “cfs” parameter is ‘all’ or a list of custom field IDs

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
<th>Data type</th>
<th>Notes on value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CF (Custom Field Name)</strong></td>
<td>A single column per custom field requested.</td>
<td><em>variable</em></td>
<td></td>
</tr>
</tbody>
</table>
Table 4 - Example tab-delimited transaction data

<table>
<thead>
<tr>
<th>Row</th>
<th>Anon Student Id</th>
<th>Session Id</th>
<th>Time</th>
<th>Duration (sec)</th>
<th>Student Response Type</th>
<th>Tutor Response Type</th>
<th>Problem Name</th>
<th>Step Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>D8DD5</td>
<td>D8DD5- Jan07-15-12-09</td>
<td>2008-01-07 15:12:18.0</td>
<td>ATTEMPT</td>
<td>RESULT</td>
<td>CHARGE1A</td>
<td>(CHOOSE-ANSWER MC-1 1)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>D8DD5</td>
<td>D8DD5- Jan07-15-12-09</td>
<td>2008-01-07 15:12:45.0</td>
<td>ATTEMPT</td>
<td>RESULT</td>
<td>CHARGE1A</td>
<td>(CHOOSE-ANSWER MC-2 2)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attempt At Step</th>
<th>Outcome</th>
<th>Selection</th>
<th>Input</th>
<th>Feedback Text</th>
<th>Feedback Classification</th>
<th>KC (Default)</th>
<th>KC Category (Default)</th>
<th>KC (Unique-step)</th>
<th>KC Category (Unique-step)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CORRECT</td>
<td>MC-1</td>
<td>1</td>
<td></td>
<td>SELECT-MC-ANSWER</td>
<td>KC1797</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>CORRECT</td>
<td>MC-2</td>
<td>2</td>
<td></td>
<td>SELECT-MC-ANSWER</td>
<td>KC1241</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
6.4  Get Student-Step Records

http://pslcdatashop.web.cmu.edu/services/datasets/[id]/[?samples/id]/steps
Returns zero or more student-step records in tab-delimited format, optionally zipped. If a sample is not specified then the “All Data” sample is returned. Note that the specified dataset must be released for you to get its student-step records.

See “Interacting with Cached Data” note above.

6.4.1  Request Parameters

zip
   true or false. Default is false. If true, the data is zipped.

cols
   Optional list of column names, comma-delimited. Default is to include all columns. See Table 5 – Student-step columns below for the list of column values.

cfs [coming soon]
   all, none, [ids]. Default is all. Optionally specify whether to include all, none, or some custom fields. If all, include all custom fields. If none, no custom fields are included. If a comma-delimited list of ids, then the specified custom fields are included, e.g. cfs=123,456.

csms
   all or none. Default is all. Optionally specify whether to include all KC models or none. If all, include all KC models. If none, no KC models are included. See Table 7 below for list of columns included with each KC model. (A future release of web services will support requesting a subset of KC models by specifying a comma-delimited list of KC model IDs).

headers
   true or false. Default is true. Specify whether a header row should be included in the output. If false, a header row is omitted.

limit
   Maximum number of student-step rows to retrieve. Default is 100, maximum is 5,000.

offset
   From the beginning of the data, the number of student-step rows to skip. Default is 0. For example, an offset of 0 would return rows starting with the 1\textsuperscript{st} row, while an offset of 100 would return rows starting with the 101\textsuperscript{st} row. Loop through rows in batches by combining offset and limit. For example, to get rows in batches of 100, request \texttt{offset=0&limit=100} in the first iteration of the loop, which will return rows 1-100. Then in the next iteration of the loop, request \texttt{offset=100&limit=100}, which will return rows 101-200.
6.4.2 Example request (subset of columns):
GET
https://pslcdatashop.web.cmu.edu/services/datasets/114/samples/21/steps?cols=row,anon_student_id,condition,problem_hierarchy,problem_name,step_name,step_duration,first_attempt&kcms=all

6.4.3 Example response:
See Table 7 - Example tab-delimited step data.

6.4.4 Example response on error:
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message
   result_code="-7"
   result_message="Error. Invalid column: time_zones." />
### Table 5 – Student-step columns

<table>
<thead>
<tr>
<th>Column name in request</th>
<th>Column(s) contained in response</th>
<th>Data type</th>
<th>Notes on value</th>
</tr>
</thead>
<tbody>
<tr>
<td>row</td>
<td>Row</td>
<td>numeric</td>
<td></td>
</tr>
<tr>
<td>anon_student_id</td>
<td>Anon Student Id</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>problem_hierarchy</td>
<td>Problem Hierarchy</td>
<td>string</td>
<td>Single column (different than Get Transactions).</td>
</tr>
<tr>
<td>problem_name</td>
<td>Problem Name</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>problem_view</td>
<td>Problem View</td>
<td>numeric</td>
<td></td>
</tr>
<tr>
<td>step_name</td>
<td>Step Name</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>step_start_time</td>
<td>Step Start Time</td>
<td>date</td>
<td>yyyy-MM-dd HH:mm:ss</td>
</tr>
<tr>
<td>first_transaction_time</td>
<td>First Transaction Time</td>
<td>date</td>
<td></td>
</tr>
<tr>
<td>correct_transaction_time</td>
<td>Correct Transaction Time</td>
<td>date</td>
<td></td>
</tr>
<tr>
<td>step_end_time</td>
<td>Step End Time</td>
<td>date</td>
<td></td>
</tr>
<tr>
<td>step_duration</td>
<td>Step Duration (sec)</td>
<td>numeric</td>
<td>Null values are represented with a dot (“.”)</td>
</tr>
<tr>
<td>correct_step_duration</td>
<td>Correct Step Duration (sec)</td>
<td>numeric</td>
<td>Null values are represented with a dot (“.”)</td>
</tr>
<tr>
<td>error_step_duration</td>
<td>Error Step Duration (sec)</td>
<td>numeric</td>
<td>Null values are represented with a dot (“.”)</td>
</tr>
<tr>
<td>first_attempt</td>
<td>First Attempt</td>
<td>string</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>incorrects</td>
<td>Incorrects</td>
<td>numeric</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>hints</td>
<td>Hints</td>
<td>numeric</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>corrects</td>
<td>Corrects</td>
<td>numeric</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>condition</td>
<td>Condition</td>
<td>string</td>
<td>In the case of a student assigned to multiple conditions (factors in a factorial design), condition names are separated by a comma and space. This differs from the format in Get Transactions.</td>
</tr>
</tbody>
</table>

**Note:** For more documentation on these columns and the semantics of the format, see [Student-Step Rollup](#) as documented in the DataShop web application.
Table 6 – KC model columns included when requested using the “kcms” parameter

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
<th>Data type</th>
<th>Notes on value</th>
</tr>
</thead>
<tbody>
<tr>
<td>KC (KC Model Name)</td>
<td>In the case of multiple KCs assigned to a single step, KC names are separated by two tildes (“~~”).</td>
<td>string</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>Opportunity (KC Model Name)</td>
<td>Opportunity number for the listed KC(s). In the case of multiple KCs assigned to a single step, opportunity number values are separated by two tildes (“~~”) and are given in the same order as the KC names.</td>
<td>numeric</td>
<td>Possibly empty</td>
</tr>
<tr>
<td>Predicted Error Rate (KC Model Name)</td>
<td>Predicted error rate for the listed KC(s). In the case of multiple KCs assigned to a single step, predicted error rate values are separated by two tildes (“~~”) and are given in the same order as the KC names.</td>
<td>numeric</td>
<td>Possibly empty</td>
</tr>
</tbody>
</table>

Note: For each KC model, the three columns listed in the above table will appear in the output.

Important: The format of the KC model columns returned by Get Student-Step Records is different from the Get Transactions web service. In the Get Transactions web service, multiple KCs associated with a transaction result in multiple columns (one column per KC). In the Get Student-Step Records web service, multiple KCs are contained in a single value and delimited with two tildes (“~~”). The same rule is applied to the Opportunity and Predicted Error Rate columns.
<table>
<thead>
<tr>
<th>Row</th>
<th>Anon Student Id</th>
<th>Condition</th>
<th>Problem Hierarchy</th>
<th>Problem Name</th>
<th>Step Name</th>
<th>Step Duration (sec)</th>
<th>First Attempt</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>D8DD5</td>
<td>a</td>
<td>Unit 1, Section 2</td>
<td>CHARGE1A</td>
<td>(CHOOSE-ANSWER MC-1 1)</td>
<td>5</td>
<td>correct</td>
</tr>
<tr>
<td>2</td>
<td>D8DD5</td>
<td>a</td>
<td>Unit 1, Section 2</td>
<td>CHARGE1A</td>
<td>(CHOOSE-ANSWER MC-2 2)</td>
<td>27</td>
<td>correct</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>KC (Default)</th>
<th>Opportunity (Default)</th>
<th>Predicted Error Rate (Default)</th>
<th>KC (Unique-step)</th>
<th>Opportunity (Unique-step)</th>
<th>Predicted Error Rate (Unique-step)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELECT-MC-ANSWER~~APPLY-THEOREM</td>
<td>4~~3</td>
<td>0.553~~0.754</td>
<td>KC1797</td>
<td>2</td>
<td>0.876</td>
</tr>
<tr>
<td>SELECT-MC-ANSWER~~APPLY-THEOREM</td>
<td>5~~4</td>
<td>0.451~~0.750</td>
<td>KC1241</td>
<td>3</td>
<td>0.639</td>
</tr>
</tbody>
</table>
7 External Analyses
DataShop supports adding and deleting external analyses via web services or the web application. An external analysis is a file attached to a dataset that describes the result of an analysis on that data. In addition to the file itself, which can be of any type, the analysis also has a required title and optional fields for a description, statistical model used, and relevant KC model.

7.1 Access and ownership
A DataShop user may add an external analysis to a dataset in a project that they have “edit” or “admin” access to. The various access types are described in section 6.1.2.

To view a list of external analyses or to retrieve the content of a single analysis, the user must have “view”, “edit”, or “admin” access.

An external analysis has an owner, the user who created it. Only the owner of an external analysis or a DataShop administrator can delete the external analysis (or modify it in the web application). To delete an external analysis, the user must be both the owner of that analysis and have “edit” access.

7.2 Add External Analysis

https://pslcdatashop.web.cmu.edu/services/datasets/[id]/analyses/add
Add and describe a new external analysis. An ID is assigned by DataShop after a successful add, and is returned in the response XML. This ID can be used in other operations that reference it. This service expects text content only; binary files can be added through the web application. For the purposes of displaying in the web application, DataShop will generate a file name for the text content added through this service.

7.2.1 Request POST Parameters

The body of the request

Required. The analysis text to be stored. Since the entire body of the request will be treated as the analysis, any parameters must be passed as part of the request URL.

    title
    Required. Name for the analysis. Must be no more than 255 characters.

    description
    Description of the analysis. Must be no more than 500 characters.

    kc_model
    Valid ID of a KC model for this dataset. KC model IDs can be obtained from a verbose request for dataset metadata.

    statistical_model
    Statistical model used in the generation of this analysis. Must be no more than 100 characters.
7.2.2 Example request:
PUT https://pslcdatashop.web.cmu.edu/services/datasets/123/analyses/add?title=
Bayesian%20Knowledge%20Tracing%20model%20with%20cross-validation&kc_model=7

7.2.3 Example request using sample DataShop web services client:
C:\DS_webservices_java1.5>java -jar dist/datashop-webservices.jar
"https://pslcdatashop.web.cmu.edu/services/datasets/76/analyses/add?title=Demo%20Title" file
name_of_file

7.2.4 Example response on success:
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message
  result_code="0"
  result_message="Success."
  analysis_id="3" />

7.2.5 Example response on error:
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message
  result_code="-8"
  result_message="Error. Required field(s) missing: title." /&gt;
7.3 Get External Analyses Metadata

https://pslcdatashop.web.cmu.edu/services/datasets/[id]/analyses

Get a listing of external analyses with metadata for the specified dataset.

7.3.1 Request Parameters

*None.*

7.3.2 Example request to get all external analyses:

GET https://pslcdatashop.web.cmu.edu/services/datasets/388/analyses

7.3.3 Example response on success:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message result_code="0" result_message="Success."
<analysis id="1">
<title>Rasch (1pl IRT) model with cross-validation</title>
<description></description>
<kce_model_name>Area</kce_model_name>
<statistical_model>Rasch</statistical_model>
<file_name>ds76__Rasch.txt</file_name>
<owner>user@ANDREW.CMU.EDU</owner>
<added>2012-06-06</added>
</analysis>
</pslc_datashop_message>
```

7.3.4 Example response on error:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message
result_code="-1"
result_message="Error. Dataset ID 45 is not valid." />
```

7.4 Get External Analysis

https://pslcdatashop.web.cmu.edu/services/datasets/[id]/analyses/[id]

Get an external analysis, identified by its ID. Returns the contents of the file.

7.4.1 Parameters

*None.*

7.4.2 Example request to get an external analysis:

GET https://pslcdatashop.web.cmu.edu/services/datasets/42/analyses/3

7.4.3 Example response:

```
DATASETID:   76
DATE:        2012-06-04 10:42:46
KC MODEL:    ANY
MODEL:       Rasch
MODEL SETUP: lme4, all defaults
```
COMPUTED BY: R

Data points: 5388
loglikelihood: -2750.70090
AIC: 5507.40180
BIC: 5527.17759
RMSE: 0.39183
A': 0.77800
10-FOLD CROSS-VALIDATED RMSE USER-STRATIFIED: 0.41557
10-FOLD CROSS-VALIDATED RMSE STEP-STRATIFIED: 0.43114
10-FOLD CROSS-VALIDATED RMSE UN-STRATIFIED: 0.40701
10-FOLD CROSS-VALIDATED A’ USER-STRATIFIED: 0.68308
10-FOLD CROSS-VALIDATED A’ STEP-STRATIFIED: 0.58122
10-FOLD CROSS-VALIDATED A’ UN-STRATIFIED: 0.72249
...

7.4.4 Example response on error:
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message
  result_code="-9"
  result_message="Error. External analysis 3 is not valid for dataset 42." />

7.5 Delete External Analysis

https://pslcdatashop.web.cmu.edu/services/datasets/[id]/analyses/[id]/delete
Remove an external analysis. You must have “edit” or “admin” access to the dataset and be the owner of the external analysis.

No request parameters. No request text content; URL serves as the space for the parameters.

7.5.1 Example request to delete an external analysis:
DELETE https://pslcdatashop.web.cmu.edu/services/datasets/42/analyses/3/delete

7.5.2 Example response on success:
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message
  result_code="0"
  result_message="Success. External analysis 3 successfully deleted." />

7.5.3 Example response on error:
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message
  result_code="-12"
  result_message="Error. Insufficient privileges to delete external analysis 3. You are not the owner." />
8 Custom Fields

DataShop supports adding and modifying custom fields at the transaction level. This section describes custom fields and how you can use them, and includes the API for creating, retrieving, and modifying custom fields through web services.

A custom field is a new column you define for annotating transaction data. Although the feature is new to web services, some datasets in DataShop already have custom fields. This is because some tutors have been instrumented to record custom fields while logging. At logging time, custom fields can be associated with a context, tool, or tutor message.

Some examples include a field that captures the time of each tutor response to the millisecond; a field noting the agent that took the action in a multi-agent system; and a field recording a categorization of the problem the student is working on.

8.1 Access and ownership

A custom field has an owner, the user who created it. Users who have edit or admin permission for a project can create custom fields for a dataset in it. Only the owner, project administrator, or a DataShop administrator can delete, modify the meta data, or set the values for the custom field. Only DataShop administrators can delete custom fields that were logged with the data.

8.2 Data Types

A custom field can be of one of the following data types:

<table>
<thead>
<tr>
<th>Data type</th>
<th>Notes on value</th>
</tr>
</thead>
<tbody>
<tr>
<td>number</td>
<td>Must be an integer or float. An integer is a number without a decimal place, while a float is a floating point number, which means it has a decimal place.</td>
</tr>
<tr>
<td>string</td>
<td>Must be no more than 65,000 characters.</td>
</tr>
<tr>
<td>date</td>
<td>Must be formatted as yyyy-MM-dd HH:mm:ss</td>
</tr>
</tbody>
</table>

Data type is reported in the metadata about the custom field when it is created or retrieved.

8.3 Add Custom Field

https://pslcdatashop.web.cmu.edu/services/datasets/[id]/customfields/add

Add and describe a new custom field. An ID is assigned by DataShop after a successful add, and is returned in the response XML. This ID can be used in other operations that reference it.

8.3.1 Request Parameters

None.
8.3.2  Request Post Parameters

All fields except description are required.

- **name**: descriptive name for the new custom field. Must be unique across all custom fields for the dataset. Must be no more than 255 characters.
- **description**: description for the new custom field. Must be no more than 500 characters.
- **level**: the level of aggregation that the custom field describes. The only accepted value for the current version of this web service is **transaction**. Future versions may support other levels such as **step** or **student**. Cannot be modified later.

8.3.3  Example request using sample DataShop web services client:
C:\DS_webservices_java1.5>java -jar dist/datashop-webservices.jar
"https://pslcdatashop.web.cmu.edu/services/datasets/123/customfields/add" file name_of_file

8.3.4  Example request:
POST https://pslcdatashop.web.cmu.edu/services/datasets/123/customfields/add

```xml
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message>
  <custom_field>
    <name>pBoredom</name>
    <description>Probability the student is bored at this transaction, calculated using Ryan Baker’s boredom detector.</description>
    <level>transaction</level>
  </custom_field>
</pslc_datashop_message>
```

8.3.5  Example response on success:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message
  result_code="0"
  result_message="Success." custom_field_id="211" />
```

8.3.6  Example response on error:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message
  result_code="-8"
  result_message="Error. Required field(s) missing: Name." />
```

See Appendix A for a full list of possible errors.
8.4 Get Custom Field Metadata

https://pslcdatashop.web.cmu.edu/services/datasets/[id]/customfields/[?id]
Get a listing of custom fields with metadata for the specified dataset, or about just a single custom field.

8.4.1 Request Parameters

mine
true or false. Default is false. If true, only retrieve metadata for custom fields you own.

8.4.2 Example request to get all custom fields:
GET https://pslcdatashop.web.cmu.edu/services/datasets/388/customfields

8.4.3 Example request to get a single custom field:
GET https://pslcdatashop.web.cmu.edu/services/datasets/388/customfields/45

8.4.4 Example response on success:
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message result_code="0" result_message="Success.">
  <custom_field id="45">
    <name>pBoredom</name>
    <description>Probability the student is bored at this transaction, calculated using Ryan Baker's boredom detector.</description>
    <level>transaction</level>
    <owner>username</owner>
    <added>2013-04-22</added>
  </custom_field>
</pslc_datashop_message>

8.4.5 Example response on error:
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message result_code="-11" result_message="Error. Custom field 45 not found."

8.5 Set Custom Field

https://pslcdatashop.web.cmu.edu/services/datasets/[id]/customfields/[id]/set
Annotate transaction data with custom fields. The custom field must first be created using the Add Custom Field method. Supports annotation for a single custom field across transactions, so to annotate a transaction with multiple custom fields, use multiple requests. This can only be done by the custom-field owner, project administrator, or a DataShop administrator on user-created custom fields.

8.5.1 Request Parameters
None.

8.5.2 Example request (tab-delimited):

<table>
<thead>
<tr>
<th>Transaction Id</th>
<th>value</th>
</tr>
</thead>
</table>
8.5.3 Example request using sample DataShop web services client:
C:\DS_webservices_java1.5> java -jar dist/datashop-webservices.jar
"https://pslcdatashop.web.cmu.edu/services/datasets/123/customfields/211/set" file name_of_file

8.5.4 Example response on success:
```xml
<pslc_datashop_message
   result_code="0"
   result_message="Success. Annotated 2 transactions." />
```

8.5.5 Example response on error:
```xml
<pslc_datashop_message
   result_code="-10"
   result_message="Error. Invalid data." />
```

8.6 Delete Custom Field

https://pslcdatashop.web.cmu.edu/services/datasets/[id]/customfields/[id]/delete
Remove a custom field entirely, the metadata and all the values associated with transactions. This can only be done by the custom-field owner, project administrator (user-created custom fields), or a DataShop administrator.

No request parameters. No request text content; URL serves as the space for the parameters.

8.6.1 Example response on success:
```xml
<pslc_datashop_message
   result_code="0"
   result_message="Success. Custom field successfully removed from 1,231 transactions." />
```

8.6.2 Example response on error:
```xml
<pslc_datashop_message
   result_code="-11"
   result_message="Error. Custom field 2 not found." />
```
9 Learning Curve

DataShop supports learning curve categorization of Knowledge Components (KCs). This section describes the categorizations and how they can be retrieved through web services.

In the DataShop web application learning curves for a dataset and skill model are placed in one of four categories: Too Little Data, Low and Flat, No Learning and Still High. Learning curves that do not fall into any of the above "bad" or "at risk" categories are labeled “Good” as they appear to indicate substantial student learning.

The algorithm for categorizing KCs first discards points in each curve based on the student threshold. If a point has fewer than that number of students, it is ignored. Within the points of curve remaining:

- If the number of points is below the opportunity threshold, then that curve has too little data.
- If all points of the curve are beneath the low error threshold, then the curve is low and flat.
- If the slope of the predicted learning curve (as determined by the AFM algorithm) is below the AFM slope threshold, then the curve shows no learning.
- If the last point of the curve is above the high error threshold, then the curve is still high.

As with the web application, the user must specify the dataset in generating the web services learning curve classification report. The user can optionally specify a skill model; the default behavior will include all skill models for the dataset in the report. The above thresholds have default values or the user may specify them and override the default value. The output is in plain text, tab-delimited format. In addition to the learning curve category, the report also includes the KC intercept, KC slope, unique step count and step instance count.

9.1 Access

Any user who has view right to a dataset can use this web service.

9.2 Get Learning Curve

https://psledatashop.web.cmu.edu/services/datasets/[id]/learningcurves/classify[?kc_model=modelName]

Get a listing of learning curve categories for all skills for the specified dataset and skill model.

9.2.1 Request Parameters

kc_model
Not required. Default all KCMs for the dataset. Case sensitive.

opportunity_threshold
Not required. Default 3. Any graph point that has lower than this opportunity count is eliminated.

student_threshold
Not required. Default 10. Any graph point that has lower than this student count is eliminated.
low_error_threshold
Not required. Default 20. Any graph point that has lower than this error rate is eliminated.

high_error_threshold
Not required. Default 40. Any graph point that has higher than this error rate is eliminated.

AFM_slope_threshold
Not required. Default 0.001. Any graph point that has lower than this slope is eliminated.

9.2.2 Example request with default:

GET
https://pslcdatashop.web.cmu.edu/services/datasets/[id]/learningcurve/classify[?kc_model=Area]

9.2.3 Example request with specified parameter:

GET
https://pslcdatashop.web.cmu.edu/services/datasets/[id]/learningcurve/classify[?kc_model=Area&opportunity_threshold=1&student_threshold=2]

9.2.4 Example response on success:

Dataset: Geometry Area (1996-97)
AFM slope threshold: 0.001
Student threshold: 10
Opportunity threshold: 3
Low error threshold: 20.0
High error threshold: 40.0

<table>
<thead>
<tr>
<th>KC Model</th>
<th>KC Name</th>
<th>Category</th>
<th>KC Intercept</th>
<th>KC Slope</th>
<th># unique steps</th>
<th># opportunity 1 step instances</th>
<th># step instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>Area formula</td>
<td>No learning</td>
<td>0.959860130611057</td>
<td>0.0</td>
<td>11</td>
<td>50</td>
<td>1784</td>
</tr>
<tr>
<td>Area</td>
<td>Non-area formula</td>
<td>Good</td>
<td>0.9359422484296835</td>
<td>0.0018665230448641058</td>
<td>22</td>
<td>59</td>
<td>3083</td>
</tr>
<tr>
<td>Geometry</td>
<td>Geometry</td>
<td>No learning</td>
<td>0.9690713757580114</td>
<td>2.226319206785172E-4</td>
<td>22</td>
<td>59</td>
<td>4843</td>
</tr>
</tbody>
</table>

Summary

<table>
<thead>
<tr>
<th>KC Model</th>
<th>%Good</th>
<th>%No learning</th>
<th>%Low and flat</th>
<th>%Still high</th>
<th>%Too little data</th>
<th># observation</th>
<th>AIC</th>
<th>Item CV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>50</td>
<td>50</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5104</td>
<td>0.408165</td>
<td></td>
</tr>
<tr>
<td>Geometry</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5104</td>
<td>0.408165</td>
<td></td>
</tr>
</tbody>
</table>

9.2.5 Example response on error:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message result_code="-1" result_message="Error. Dataset 9 is not valid."/>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<pslc_datashop_message result_code="-18" result_message="Error. Skill model area is not valid."/>
```
## Appendix A

### All Possible Result Codes

<table>
<thead>
<tr>
<th>Result Code</th>
<th>HTTP Status Code</th>
<th>Description</th>
<th>Services that could return this result code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>200 OK</td>
<td>Success. [Optional message]</td>
<td>All</td>
</tr>
<tr>
<td>-1</td>
<td>404 Not Found</td>
<td>Error. Dataset [id] is not valid.</td>
<td>All</td>
</tr>
<tr>
<td>-2</td>
<td>401 Unauthorized</td>
<td>Error. Dataset [id] is not accessible.</td>
<td>All</td>
</tr>
<tr>
<td>-3</td>
<td>404 Not Found</td>
<td>Error. Sample [id] is not valid for dataset [id].</td>
<td>Get/Samples, Get Transactions, Get Student-Steps</td>
</tr>
<tr>
<td>-4</td>
<td>401 Unauthorized</td>
<td>Error. Sample [id] is not accessible for dataset [id].</td>
<td>Get/Samples, Get Transactions, Get Student-Steps</td>
</tr>
<tr>
<td>-5</td>
<td>400 Bad Request</td>
<td>Error. Invalid request parameter: [parameter].</td>
<td>All</td>
</tr>
<tr>
<td>-6</td>
<td>400 Bad Request</td>
<td>Error. Invalid value for parameter [param]: [value].</td>
<td>All</td>
</tr>
<tr>
<td>-7</td>
<td>401 Unauthorized</td>
<td>Error. Dataset [id] is not released.</td>
<td>Get Transactions, Get Student-Steps</td>
</tr>
<tr>
<td>-8</td>
<td>400 Bad Request</td>
<td>Error. Required field(s) missing: [fields].</td>
<td>Add External Analysis, Add Custom Field, Get Learning Curve</td>
</tr>
<tr>
<td>-9</td>
<td>404 Not Found</td>
<td>Error. [External analysis / Custom field] [id] is not valid for dataset [id].</td>
<td>Get/delete External Analysis, All Custom Field services</td>
</tr>
<tr>
<td>-10</td>
<td>400 Bad Request</td>
<td>Error. Invalid data.</td>
<td>Add External Analysis, Set Custom Field</td>
</tr>
<tr>
<td>-11</td>
<td>404 Not Found</td>
<td>Error. Custom field [id] not found.</td>
<td>Delete Custom Field</td>
</tr>
<tr>
<td>-12</td>
<td>401 Unauthorized</td>
<td>Error. Insufficient privileges to [add/delete/set] [an/a] [external analysis / custom field] [id]. [You are not the owner.]</td>
<td>Delete External Analysis, Delete Custom Field, Add External Analysis, Add Custom Field</td>
</tr>
<tr>
<td>-13</td>
<td>409 Conflict</td>
<td>Error. Custom field with name [name] already exists for this dataset.</td>
<td>Add Custom Field</td>
</tr>
<tr>
<td>-14</td>
<td>404 Not Found</td>
<td>Error. [Transactions/Student-steps] cache file for sample [N] [does not exist / is out-of-date]. Caching process is starting. Try again later.</td>
<td>Get Transactions, Get Student-Steps</td>
</tr>
<tr>
<td>-15</td>
<td>400 Bad Request</td>
<td>Error. Parameter [parameter] must be</td>
<td>Add external analysis</td>
</tr>
</tbody>
</table>
no more than [N] characters.

<table>
<thead>
<tr>
<th>Code</th>
<th>Status Code</th>
<th>Message</th>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>-16</td>
<td>400 Bad Request</td>
<td>Error. Invalid value for element '[element]': [value]</td>
<td>Add Custom Field</td>
</tr>
<tr>
<td>-17</td>
<td>400 Bad Request</td>
<td>Invalid XML format.</td>
<td>Add Custom Field</td>
</tr>
<tr>
<td>-18</td>
<td>404 Not Found</td>
<td>Error. Skill model name [name] is not valid.</td>
<td>Get Learning Curve</td>
</tr>
<tr>
<td>-99</td>
<td>404 Not Found</td>
<td>Error. No web service found matching the URL. For a list of valid URLs, see <a href="http://pslcdatashop.org/api/">http://pslcdatashop.org/api/</a></td>
<td>All</td>
</tr>
<tr>
<td>-100</td>
<td>500 Internal Server Error</td>
<td>Unknown error.</td>
<td>All</td>
</tr>
<tr>
<td>-101</td>
<td>401 Unauthorized</td>
<td>Authorization failed. Check your credentials.</td>
<td>All</td>
</tr>
<tr>
<td>-102</td>
<td>406 Not Acceptable</td>
<td>This content is available only as [content type].</td>
<td>All</td>
</tr>
<tr>
<td>-103</td>
<td>405 Method Not Allowed</td>
<td>Operation not supported. (Note to reader: this is for unsupported PUT/POST/DELETE methods.)</td>
<td>All</td>
</tr>
<tr>
<td>-104</td>
<td>405 Method Not Allowed</td>
<td>[Method] requests not supported. (Note to reader: this is for methods head, options, etc.)</td>
<td>All</td>
</tr>
</tbody>
</table>
### Appendix B

#### Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date Released</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revision 0.3</td>
<td>October 24, 2013</td>
<td>Added section “Custom Fields” and updated the result codes table.</td>
</tr>
<tr>
<td>Revision 0.23</td>
<td>October 11, 2012</td>
<td>Added section “External Analyses” and updated the result codes table.</td>
</tr>
<tr>
<td>Revision 0.22</td>
<td>December 17, 2010</td>
<td>Added new cross-validation elements and number of parameters to example XML for Get dataset metadata (verbose). Also changed element “Ifa_status” to “logistic_regression_model_status” in the same section.</td>
</tr>
<tr>
<td>Revision 0.21</td>
<td>June 29, 2010</td>
<td>Added “domain” element to Get Dataset Metadata examples as web services now include a “domain” element (if it’s set) in addition to “learnlab”.</td>
</tr>
</tbody>
</table>