Statistical Tests

A **statistical test** resource automatically runs some advanced statistical tests on the numeric fields of a dataset. The goal of these tests is to check whether the values of individual fields conform or differ from some distribution patterns. Statistical test are useful in tasks such as fraud, normality, or outlier detection.

The tests are grouped in the following three categories:

- **Fraud Detection Tests**:

  - Benford: This statistical test performs a comparison of the distribution of first significant digits (FSDs) of each value of the field to the Benford's law distribution. Benford's law applies to numerical distributions spanning several orders of magnitude, such as the values found on financial balance sheets. It states that the frequency distribution of leading, or first significant digits (FSD) in such distributions is not uniform. On the contrary, lower digits like 1 and 2 occur disproportionately often as leading significant digits. The test compares the distribution in the field to Bendford's distribution using a Chi-square goodness-of-fit test, and Cho-Gaines d test. If a field has a dissimilar distribution, it may contain anomalous or fraudulent values.

- **Normality tests**: These tests can be used to confirm the assumption that the data in each field of a dataset is distributed according to a normal distribution. The results are relevant because many statistical and machine learning techniques rely on this assumption.

  - Anderson-Darling: The Anderson-Darling test computes a test statistic based on the difference between the observed cumulative distribution function (CDF) to that of a normal distribution. A significant result indicates that the assumption of normality is rejected.

  - Jarque-Bera: The Jarque-Bera test computes a test statistic based on the third and fourth central moments (skewness and kurtosis) of the data. Again, a significant result indicates that the normality assumption is rejected.

  - Z-score: For a given sample size, the maximum deviation from the mean that would expected in a sampling of a normal distribution can be computed based on the 68-95-99.7 rule. This test simply reports this expected deviation and the actual deviation observed in the data, as a sort of sanity check.

- **Outlier tests**:

  - Grubbs: When the values of a field are normally distributed, a few values may still deviate from the mean distribution. The outlier tests reports whether at least one value in each numeric field differs significantly from the mean using Grubb's test for outliers. If an outlier is found, then its value will be returned.

Note that both the number of tests within each category and the categories may increase in the near future.

BigML.io allows you to create, retrieve, update, and delete your **statistical test**. You can also list all of your

statistical tests.

Jump to:

Statistical Test Base URL

You can use the following base URL to create, retrieve, update, and delete **statistical tests**.

```bash
https://bigml.io/andromeda/statisticaltest
```

All requests to manage your **statistical tests** must use HTTPS and be authenticated using your **username** and **API key** to verify your identity. See this section for more details.

Creating a Statistical Test

To create a new **statistical test**, you need to POST to the **statistical test** base URL an object containing at least the **dataset/id** that you want to use to create the **statistical test**. The **content-type** must always be **"application/json"**.

You can easily create a new **statistical test** using **curl** as follows. All you need is a valid **dataset/id** and your authentication variable set up as shown above.

```curl
curl "https://bigml.io/andromeda/statisticaltest?$BIGML_AUTH" \
    -X POST \
    -H 'content-type: application/json' \
```

```
          -d '{"dataset": "dataset/603e20a91f386f43db000004"}'
```

BigML.io will return a newly created **statistical test** document, if the request succeeded.

```json
{
    "category": 0,
    "code": 201,
    "columns": 5,
    "configuration": null,
    "configuration_status": false,
    "created": "2021-03-03T08:45:57.525334",
    "creator": "alfred",
    "dataset": "dataset/603e20a91f386f43db000004",
    "dataset_field_types": {
        "auto_generated": {},
        "categorical": 1,
        "datetime": 0,
        "image": 0,
```

Statistical Test Arguments

In addition to the **dataset**, you can also POST the following arguments.

Argument	Type	Description
ad_sample_size optional	Integer, default is **1024**	The Anderson-Darling normality test is computed from a sample from the values of each field. This parameter specifies the number of samples to be used during the normality test. If not given, defaults to 1024. **Example**: 128
ad_seed optional	String	A string to be hashed to generate deterministic samples for the Anderson-Darling normality test. **Example**: "MyADSeed"
category optional	Integer, default is **the category of**	The category that best describes the **statistical test**. See the category codes for the complete list of categories.

Argument	Type	Description
	the dataset	**Example**: 1
dataset	String	A valid **dataset/id**. **Example**: dataset/603e20a91f386f43db000004

datasets
optional
	Array	A list of dataset ids or objects to be used to build the new **statistical test**. See the Section on Multi-Datasets and Section on Resources Accepting Multi-Datasets Input for more details.
		Example:
		[
		{
		 "id": "dataset/603e20a91f386f43db000004",
		 "sample_rate": 0.5,
		 "out_of_bag": true
		},
		{
		 "id": "dataset/52bc851b3c1920e4a3000022",
		 "sample_rate": 0.8,
		 "replacement": true
		}
		]

default_numeric_value optional	String	It accepts any of the following strings to substitute missing numeric values across all the numeric fields in the dataset: **mean**, **median**, **minimum**, **maximum**, **zero** **Example**: "mean"
description optional	String	A description of the **statistical test** up to 8192 characters long. **Example**: "This is a description of my new statistical test"
excluded_fields optional	Array, default is **[],** **an empty list.** **None of the** **fields in the** **dataset is** **excluded.**	Specifies the fields that won't be included in the **statistical test**. **Example**: ["000000", "000002"]

Argument	Type	Description
fields optional	Object, default is **{}, an empty dictionary. That is, no names or preferred statuses are changed**.	This can be used to change the names of the fields in the statistical test with respect to the original names in the dataset or to tell **BigML** that certain fields should be preferred. An entry keyed with the field id generated in the **source** for each field that you want the name updated. **Example**: { "000001": {"name": "length_1"}, "000003": {"name": "length_2"}, }
fields_maps optional	Object	A dictionary keyed by **dataset/id** and **object** values. Each entry maps fields in the first dataset to fields in the dataset referenced by the key. **Example**: { "dataset/603e20a91f386f43db000004": { "000000":"000023", "000001":"000024", "000002":"00003a"}, "dataset/52bc80233c1920e4a300001a": { "000000":"000023", "000001":"000004", "000002":"00000f" }
include_extracted_features optional	Boolean or Array of IDs	Extracted image features to use as model inputs. Available options are: • **true**: include all extracted features, unless explicitly excluded (this is the default for all non-deepnet models.) • **false**: don't include any extracted features (this is the default for deepnet models, if not given.) • **list of ids**: a explicit list of field ids, corresponding to extracted fields to add to the default set **Example**: true
	Array,	

Argument	Type	Description
input_fields optional	default is **[]**. **All the fields in the dataset**.	Specifies the fields to be considered to create the **statistical test**. **Example**: ["000001", "000003"]
name optional	String default is **dataset's name**	The name you want to give to the new **statistical test**. **Example**: "my new statistical test"
out_of_bag optional	Boolean, default is **false**	Setting this parameter to **true** will return a sequence of the out-of-bag instances instead of the sampled instances. See the Section on Sampling for more details. **Example**: true
project optional	String	The **project/id** you want the **statistical test** to belong to. **Example**: "project/603de73d1f386f7360000000"
range optional	Array, default is **[1, max rows in the dataset]**	The range of successive instances to build the **statistical test**. **Example**: [1, 150]
replacement optional	Boolean, default is **false**	Whether sampling should be performed with or without replacement. See the Section on Sampling for more details. **Example**: true
sample_rate optional	Float, default is **1.0**	A real number between 0 and 1 specifying the sample rate. See the Section on Sampling for more details. **Example**: 0.5
seed optional	String	A string to be hashed to generate deterministic sample. See the Section on Sampling for more details. **Example**: "MySample"
		The **shared hash** of the shared model to be cloned.

Argument	Type	Description
shared_hash	String	Set **deep** to **true** to clone the dataset used to build the statistical test too. Note that the dataset can be cloned only if it is already shared and set clonable. If multiple datasets have been used to create the statistical test, only the first dataset will be cloned. **Example**: "kpY46mNuNVReITw0Z1mAqoQ9ySW"
significance_levels optional	Array, default is **[0.01, 0.05, 0.1]**	An array of significance levels between 0 and 1 to test against p_values. **Example**: [0.01, 0.025, 0.05, 0075, 0.1]
tags optional	Array of Strings	A list of strings that help classify and index your **statistical test**. **Example**: ["best customers", "2021"]
webhook optional	Object	A webhook url and an optional secret phrase. See the Section on Webhooks for more details. **Example**: { "url": "http://myhost/path/to/webhook", "secret": "mysecret" }

You can also use **curl** to customize a new **statistical test**. For example, to create a new **statistical test** named "my statistical test", with only certain rows, and with only three fields:

```curl
curl "https://bigml.io/andromeda/statisticaltest?$BIGML_AUTH" \
    -X POST \
    -H 'content-type: application/json' \
    -d '{"dataset": "dataset/603e20a91f386f43db000004",
        "input_fields": ["000001", "000002", "000003"],
        "name": "my statistical test",
        "range": [25, 125]}'
```

If you do not specify a name, **BigML.io** will assign to the new **statistical test** the **dataset**'s name. If you do not specify a **range** of instances, **BigML.io** will use all the instances in the **dataset**. If you do not specify any **input fields**, **BigML.io** will include all the input fields in the **dataset**.

Read the Section on Sampling Your Dataset to leran how to sample your **dataset**. Here's an example of **statistical test** request with range and sampling specifications:

```curl
curl "https://bigml.io/andromeda/statisticaltest?$BIGML_AUTH" \
     -X POST \
     -H 'content-type: application/json' \
     -d '{"dataset": "dataset/603e20a91f386f43db000004",
          "range": [1, 5000],
          "sample_rate": 0.5}'
```

Retrieving a Statistical Test

Each **statistical test** has a unique identifier in the form **"statisticaltest/id"** where id is a string of 24 alpha-numeric characters that you can use to retrieve the **statistical test**.

To retrieve a **statistical test** with **curl**:

```curl
curl "https://bigml.io/andromeda/statisticaltest/603f3eb91f386fa1cf000000?$BIGML_AUTH"
```

Statistical Test Properties

Once a **statistical test** has been successfully created it will have the following properties.

Property	Type	Description
category filterable, sortable, updatable	Integer	One of the categories in the table of categories that help classify this resource according to the domain of application.
code	Integer	One of the HTTP status code. This will be 201 upon successful creation of the **statistical test** and 200 afterwards. Make sure that you check the code that comes with the status attribute to make sure that the **statistical test** creation has been completed without errors.

Property	Type	Description
columns filterable, sortable	Integer	The number of fields in the **statistical test**.
created filterable, sortable	ISO-8601 Datetime	This is the date and time in which the **statistical test** was created with microsecond precision. It follows this pattern yyyy-MM-ddThh:mm:ss.SSSSSS. All times are provided in Coordinated Universal Time (UTC).
creator	String	The user that created the **statistical test**.
dataset filterable, sortable	String	The **dataset/id** that was used to build the **statistical test**.

dataset_field_types	Object	A dictionary with an entry per field type in the **dataset** and the total number of fields of that type.
		Example:
		{
		  "auto_generated": {},
		  "categorical": 1,
		  "datetime": 0,
		  "image": 0,
		  "items": 0,
		  "numeric": 4,
		  "path": 0,
		  "regions": 0,
		  "text": 0,
		  "total": 5
		}

dataset_status filterable, sortable	Boolean	Whether the **dataset** is still available or has been deleted.
datasets	Array	A list of dataset ids or objects used to build the **statistical test**.
description updatable	String	A text describing the **statistical test**. It can contain restricted markdown to decorate the text.
excluded_fields	Array	The list of **fields**'s ids that were excluded to build the **statistical test**.
		A dictionary with meta information about the fields dictionary. It

Property	Type	Description
fields_meta	Object	specifies the **total** number of fields, the current **offset**, and **limit**, and the number of fields (**count**) returned.
input_fields	Array	The list of **input fields'** ids used to build the models of the **statistical test**.
locale	String	The **dataset**'s locale.
max_columns filterable, sortable	Integer	The total number of fields in the **dataset** used to build the **statistical test**.
max_rows filterable, sortable	Integer	The maximum number of instances in the **dataset** that can be used to build the **statistical test**.
name filterable, sortable, updatable	String	The name of the **statistical test** as your provided or based on the name of the **dataset** by default.
name_options filterable, sortable	String	Information about the **statistical test**.
out_of_bag filterable, sortable	Boolean	Whether the out-of-bag instances were used to create the **statistical test** instead of the sampled instances.
price filterable, sortable, updatable	Float	The price other users must pay to clone your **statistical test**.
private filterable, sortable, updatable	Boolean	Whether the **statistical test** is public or not.
project filterable, sortable, updatable	String	The **project/id** the resource belongs to.
range	Array	The **range** of instances used to build the **statistical test**.
replacement filterable, sortable	Boolean	Whether the instances sampled to build the **statistical test** were selected using replacement or not.

Property	Type	Description
resource	String	The **statisticaltest/id**.
rows filterable, sortable	Integer	The total number of instances used to build the **statistical test**.
sample_rate filterable, sortable	Float	The sample rate used to select instances from the **dataset** to build the **statistical test**.
seed filterable, sortable	String	The string that was used to generate the sample.
shared filterable, sortable,updatable	Boolean	Whether the **statistical test** is shared using a private link or not.
shared_clonable filterable, sortable,updatable	Boolean	Whether the shared **statistical test** can be cloned or not.
shared_hash	String	The hash that gives access to this **statistical test** if it has been shared using a private link.
sharing_key	String	The alternative key that gives read access to this **statistical test**.
size filterable, sortable	Integer	The number of bytes of the **dataset** that were used to create this **statistical test**.
source filterable, sortable	String	The **source/id** that was used to build the **dataset**.
source_status filterable, sortable	Boolean	Whether the **source** is still available or has been deleted.
statistical_tests	Object	All the information that you need to recreate the **statistical test**. It includes the **field**'s dictionary describing the fields and their summaries, and the **statistical tests**. See the Statistical Tests Object definition below.

Property	Type	Description
status	Object	A description of the status of the **statistical test**. It includes a code, a message, and some extra information. See the table below.
subscription filterable, sortable	Boolean	Whether the **statistical test** was created using a subscription plan or not.
tags filterable, updatable	Array of Strings	A list of user tags that can help classify and index this resource.
updated filterable, sortable	ISO-8601 Datetime	This is the date and time in which the **statistical test** was updated with microsecond precision. It follows this pattern yyyy-MM-ddThh:mm:ss.SSSSSS. All times are provided in Coordinated Universal Time (UTC).
webhook	Object	A webhook url and an optional secret phrase. See the Section on Webhooks for more details.
white_box filterable, sortable	Boolean	Whether the **statistical test** is publicly shared as a white-box.

Statistical Test Object

The **Statistical Tests Object** of statistical test has the following properties. Many statistical tests will contain a p-value and a **significant** boolean array, indicating whether the p_value is less than the provided **significance_levels** (by default, [0.01, 0.05, 0.10] is used if not provided). If p-value is greater than the accepted significance level, then then it fails to reject the null hypothesis, meaning there is no statistically significant difference between the treatment groups. For example, if the **significance levels** is [0.01, .0.025, 0.05, 0.075, 0.1] and **p-value** is 0.05, then **significant** is [false, false, false, true, true].

Property	Type	Description
ad_sample_size	Integer	The sample test size used for the Anderson-Darling normality test.
ad_seed	String	A seed used to generate deterministic samples for the Anderson-Darling normality test.
		A dictionary with an entry per field in the dataset used to build the test. Fields are paginated according to the **field_meta** attribute. Each entry

Property	Type	Description
fields	Object	includes the column number in original source, the name of the field, the type of the field, and the **summary**. See this Section for more details.
fraud	Array	An array of anomalous fields detection test results for each numeric field. See Fraud Object.
normality	Array	An array of data normality test results for each numeric field. See Normality Object.
outliers	Array	An array of outlier detection test results for each numeric field. See Outliers Object.
significance_levels	Array	An array of user provided significance levels to test against p_values.

Fraud Object

The **Fraud Object** has the following properties.

Property	Type	Description
name	String	Name of the fraud test. Currently only value available is **benford**.
result	Object	A test result which is a dictionary between field ids and test result. The type of result object varies based on the name of the test. When **name** is **benford**, it returns Benford Result Object.

Benford Result Object

The **Benford Result Object** has the following properties. Benford's Law is a simple yet powerful tool allowing quick screening of data for anomalies.

Property	Type	Description
chi_square	Object	See Chi-Square Object.
cho_gaines	Object	See Cho-Gaines Object.

Property	Type	Description
distribution	Array	The distribution of first significant digits (FSDs) to the Benford's law distribution. For example, the FSD for 2015 is 2, and for 0.00609 is 6. The array represents the number of occurences for each digit from 1 to 9. **Example**: [0, 0, 0, 22, 61, 54, 0, 0, 0]
negatives	Integer	The number of negative values.
zeros	Integer	The number of values exactly equal to 0.

Chi-Square Object.

The **Chi-Square Object** contains the chi-square statistic used to investigate whether distributions of categorical variables differ from one another. This test is used to compare a collection of categorical data with some theoretical expected distribution. The object has the following properties.

Property	Type	Description
chi_square_value	Float	The value of the chi-square statistic. **Example**: 1201.60468
p_value	Float	A function used in the context of null hypothesis testing in order to quantify the idea of statistical significance of evidence. **Example**: 0.015
significant	Array	A boolean array indicating whether the test produced a significant result at each of the **significance_levels**. If **p_value** is less than the **significance_level**, then it indicates it is significant. The default **significance_levels are** [0.01, 0.05, 0.1]. **Example**: [false, true, true]

Cho-Gaines Object.

The **Cho-Gaines Object** has the following properties.

Property	Type	Description
d_statistic	Float	A value based on Euclidean distance from Benford's distribution in the 9-dimensional space occupied by any first-digit vector to test Cho-Gaines d test.
significant	Array	A boolean array indicating whether the test produced a significant result at each of the **significance_levels**. If **p_value** is less than the **significance_level**, then it indicates it is significant. It does not respect the values passed in **significance_levels**, but always use [0.01, 0.05, 0.1]. **Example**: [false, true, true]

Normality Object

The **Normality Object** has the following properties.

Property	Type	Description
name	String	Name of the normality test. Available values are **anderson_darling**, **jarque_bera**, and **z_score**.
result	Object	A test result which is a dictionary between field ids and test result. The type of result object varies based on the name of the test. When **name** is **anderson_darling**, it returns Anderson-Darling Result Object, when **jarque_bera**, Jarque-Bera Result Object, and when **z-score**, Z-Score Result Object.

Anderson-Darling Result Object

The **Anderson-Darling Result Object** has the following properties. See Anderson-Darling Test for more information.

Property	Type	Description
p_value	Float	A function used in the context of null hypothesis testing in order to quantify the idea of statistical significance of evidence. **Example**: 0.015
significant	Array	A boolean array indicating whether the test produced a significant result at each of the **significance_levels**. If **p_value** is less than the **significance_level**, then it indicates it is significant. The default **significance_levels** are [0.01, 0.05, 0.1].

Property	Type	Description
		Example: [false, true, true]

Jarque-Bera Result Object

The **Jarque-Bera Result Object** has the following properties. See Jarque-Bera Test for more information.

Property	Type	Description
p_value	Float	A function used in the context of null hypothesis testing in order to quantify the idea of statistical significance of evidence. **Example**: 0.015
significant	Array	A boolean array indicating whether the test produced a significant result at each of the **significance_levels**. If **p_value** is less than the **significance_level**, then it indicates it is significant. The default **significance_levels** are [0.01, 0.05, 0.1]. **Example**: [false, true, true]

Z-Score Result Object

The **Z-Score Object** has the following properties. A positive standard score indicates a datum above the mean, while a negative standard score indicates a datum below the mean. See z-score for more information.

Property	Type	Description
expected-max-z	Float	The expected maximum z-score for the sample size.
max-z	Float	The maximum z-score.

Outliers Object

The **Outliers Object** has the following properties.

Property	Type	Description
name	String	Name of the outlier detection test. Currently only value available is **grubbs**.

Property	Type	Description
result	Object	A test result which is a dictionary between field ids and test result. The type of result object varies based on the name of the test. When **name** is **grubbs**, it returns Grubbs Result Object.

Grubbs Result Object

The **Grubb's Test for Outliers Result Object** has the following properties. It computes a t-test based on the maximum deviation from the mean. A significant result indicates that at least one outlier is present in the data. If an outlier is found, also returns the value of the outlier. Note that this test assumes that the data are normally distributed. See Grubb's test for outliers for more information.

Property	Type	Description
outlier	Number	An outlier present in the data. It is available only when at at least of one of the boolean values in **significant** is true.
p_value	Float	A function used in the context of null hypothesis testing in order to quantify the idea of statistical significance of evidence. **Example**: 0.015
significant	Array	A boolean array indicating whether the test produced a significant result at each of the **significance_levels**. If **p_value** is less than the **significance_level**, then it indicates it is significant. The default **significance_levels** are [0.01, 0.05, 0.1]. **Example**: [false, true, true]

Statistical Test Status

Creating a **statistical test** is a process that can take just a few seconds or a few days depending on the size of the **dataset** used as input and on the workload of **BigML**'s systems. The **statistical test** goes through a number of states until its fully completed. Through the status field in the **statistical test** you can determine when the test has been fully processed and ready to be used to create predictions. These are the properties that **statistical test**'s **status** has:

Property	Type	Description
code	Integer	A status code that reflects the status of the **resource** creation. It can be any of those that are explained here.

Property	Type	Description
elapsed	Integer	Number of milliseconds that **BigML.io** took to process the **resource**.
message	String	A human readable message explaining the status.
progress	Float, between 0 and 1	How far **BigML.io** has progressed building the **resource**.

Once **statistical test** has been successfully created, it will look like:

```json
{
    "category": 0,
    "code": 200,
    "columns": 5,
    "configuration": null,
    "configuration_status": false,
    "created": "2021-03-03T08:45:57.525000",
    "creator": "alfred",
    "dataset": "dataset/603e20a91f386f43db000004",
    "dataset_field_types": {
        "auto_generated": {},
        "categorical": 1,
        "datetime": 0,
        "image": 0,
```

Filtering and Paginating Fields from a Statistical Test

A **statistical test** might be composed of hundreds or even thousands of fields. Thus when retrieving a **statistical test**, it's possible to specify that only a subset of fields be retrieved, by using any combination of the following parameters in the query string (unrecognized parameters are ignored):

Parameter	Type	Description
fields optional	Comma-separated list	A comma-separated list of **field** IDs to retrieve. **Example**: "fields=000000,000002"

Parameter	Type	Description
full optional	Boolean	If false, no information about fields is returned. **Example**: "full=false"
iprefix optional	String	A case-insensitive string to retrieve fields whose name start with the given prefix; It is possible to specify more than one iprefix by repeating the parameter, in which case the union of the results is returned. **Example**: "iprefix=INCOME"
limit optional	Integer	Maximum number of **fields** that you will get in the **fields** field. **Example**: "limit=100"
offset optional	Integer	How far off from the first **field** in your **dataset** is the first **field** in the **fields** field. **Example**: "offset=100"
order_by optional	String	Sorting criteria; possible values are **"count"**, **"max"**, **"min"**, **"name"**, and **"type"**, and their negated values (**"-count"**, **"-name"**, etc.) to specify a descending order. **Example**: "order_by=name"
prefix optional	String	A case-sensitive string to retrieve fields whose name start with the given prefix; It is possible to specify more than one prefix by repeating the parameter, in which case the union of the results is returned. **Example**: "prefix=income"

Since **fields** is a map and therefore not ordered, the returned fields contain an additional key, **order**, whose integer (increasing) value gives you their ordering. In all other respects, the source is the same as the one you would get without any filtering parameter above.

The **fields_meta** field can help you paginate fields. Its structure is as follows:

Property	Type	Description
count optional	Integer	Specifies the current number of fields in the resource.
limit optional	Integer	The maximum number of fields that will be returned in the resource.

Property	Type	Description
offset optional	Integer	The current offset in the pagination of fields.
total optional	Integer	The total number of fields in the resource.

Note that paginating fields might only be worth if you are going to deal with really wide (i.e., more than 200 fields).

Updating a Statistical Test

To update a **statistical test**, you need to PUT an object containing the fields that you want to update to the **statistical test**'s base URL. The content-type must always be: **"application/json"**. If the request succeeds, **BigML.io** will return with an HTTP 202 response with the updated **statistical test**.

For example, to update **statistical test** with a new name you can use **curl** like this:

```curl
curl "https://bigml.io/andromeda/statisticaltest/603f3eb91f386fa1cf000000?$BIGML_AUTH" \
    -X PUT \
     -H 'content-type: application/json' \
     -d '{"name": "a new name"}'
```

If you want to update **statistical test** with a new **label** and **description** for a specific field you can use **curl** like this:

```curl
curl "https://bigml.io/andromeda/statisticaltest/603f3eb91f386fa1cf000000?$BIGML_AUTH" \
    -X PUT \
     -H 'content-type: application/json' \
     -d '{"fields": {"000000": {
                     "label": "a longer name",
                     "description": "an even longer description"}}}'
```

See this section for more details.

Deleting a Statistical Test

To delete a **statistical test**, you need to issue a HTTP DELETE request to the **statisticaltest/id** to be deleted.

Using **curl** you can do something like this to delete a **statistical test**:

```
                                                                                    curl

  curl -X DELETE "https://bigml.io/andromeda/statisticaltest/603f3eb91f386fa1cf000000?$BIGML_
```

If the request succeeds you will not see anything on the command line unless you executed the command in verbose mode. Successful DELETEs will return **"204 no content"** responses with no body.

Once you delete a **statistical test**, it is permanently deleted. That is, a delete request cannot be undone. If you try to delete a **statistical test** a second time, or a **statistical test** that does not exist, you will receive a **"404 not found"** response.

However, if you try to delete a **statistical test** that is being used at the moment, then **BigML.io** will not accept the request and will respond with a **"400 bad request"** response.

See this section for more details.

Listing Statistical Tests

To list all the **statistical test**, you can use the **statistical test** base URL. By default, only the 20 most recent **statistical tests** will be returned. You can see below how to change this number using the **limit** parameter.

You can get your list of **statistical tests** using **curl**.

```
                                                                                    curl

  curl "https://bigml.io/andromeda/statisticaltest?$BIGML_AUTH"
```

See this section for more details. You can also paginate, filter, and order your **statistical tests**.

Tools

Certifications

Copyright © 2024 BigML, Inc.