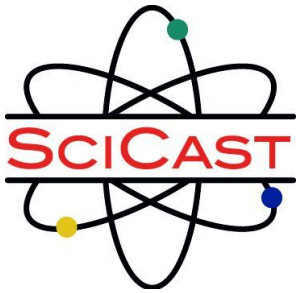

SciCast
Data Mart Guide

Version 1.2

.

23 October 2014

Table of Contents

1. Overview

Those who prefer to drive before mapping, see Quick Start.

SciCast is a science and technology combinatorial prediction market.

Forecast questions are developed in **SciCast Spark** (http://spark.scicast.org) and published and forecasted-on in **SciCast Predict** (http://www.scicast.org). Data from SciCast is made available via the **SciCast Data Mart**.

The Data Mart is a web service. It does not provide an interactive user interface. Instead, it provides a set of queries (i.e., web services) to obtain results. Web service queries can be executed via browser, command line or by writing your own script or code.

To get an API key for accessing SciCast data via the SciCast Data Mart, send a request via email to support@scicast.org. You will be emailed a URL for a Data Mart request web form. Fill out the form and push the "Submit" button. Once your information is validated you will be emailed your personal api_key. Please do not share it.

Researchers or analysts who want to do repeated data downloads may find it convenient to write scripts or a web service to execute the queries periodically. A sample script is available for re-use at https://github.com/SciCast/datamart_scripts.

The Data Mart data is updated once each 24 hours.

The Data Mart maintains histories of forecasts and marginal (unconditional) probabilities, but otherwise generally provides only current information. For example, if a question's text is modified, only the latest version of that text will be returned in the "question" Data Mart query.

Current Data Mart documentation for software engineers may be accessed at http://datamart.scicast.org/api_docs/. Much, but not all of it, is also contained in this guide. New query types and parameters will be added from time to time. These will be kept current in http://datamart.scicast.org/api_docs/, but there may be a lag in updating the Data Mart Guide.

The following sections respectively provide the web service queries and parameters, define the fields of the records returned by query execution, and explain access methods.

2. Web Service Queries and Parameters

Web service queries are structured as a base onto which parameters may be concatenated.

2.1. Web Service Query Base

The web service query base has the following structure where terms in angle brackets represent options:

http://datamart.scicast.org/<query_type>/?format=<file_type>&api_key=<api_key>

The <api_key> is emailed to you after you submit the Data Mart access request form.

The <file_type> options are:

- json
- csv
- xml

The <query_type> options are:

- comment
  - comments made on SciCast Predict
- person
  - public metadata and forecast history for each person
- person/leaderboard
  - each person on SciCast Predict in order of their highest score (expected assets) during the date interval of the query
  - if you specify only the start_date, then that specific day's leaderboard will be returned. When you also specify an end_date greater than the start_date then the highest score for each user obtained between the start_date and end_date will be returned.
- question
  - all questions on SciCast Predict, including resolved and invalid questions
- question_history
  - the time sequence of marginal probabilities on each question
- trade_history
  - the list of trades made on SciCast Predict including all data for each trade

2.2. Web Service Query Parameters

Parameters can be concatenated to queries by adding it to the request syntax. Additional parameters are separated by a "&" symbol. For example date interval parameters may be applied to any web service query as follows:

http://datamart.scicast.org/<query_type>/?format=<file_type>&api_key=<api_key>&start_date=<date1>&end_date=<date2>

where `date1` is earlier or the same as `date2`. As a parameter the dates have the form MM-DD-YYYY. (The date format for fields returned by queries may have different formats.)

An example trade_history query using the date option to download a JSON file with the trade histories for the month of December 2013 is:

http://datamart.scicast.org/trade_history/?format=json&api_key=<api_key>&start_date=12-01-2013&end_date=12-31-2013

The option "`exclude_public`" is available on all queries. This is for users whose api_key enables them access to private questions and trades. It has the options of `true` and `false`. When the exclude_public parameter is not included or is set to `false` both the public and private data is returned. When the exclude_public parameter is set to `true` only the private data is returned.

The parameter option "`aggregate_level`" is available for the `question_history` and `trade_history` queries. It has options of `daily`, `weekly`, `monthly` or `yearly`. It is concatenated to a query in the form &aggregate_level=<option>, for example:

http://datamart.scicast.org/trade_history/?format=json&api_key=<api_key>&start_date=12-01-2013&end_date=12-31-2013&aggregate_level=monthly.

When the `aggregate_level` parameter is specified on a query, statistics are also returned. These statistics are not included unless the option is included in the query.

The aggregate_level option determines the time interval over which the statistics are computed. So for example the (number of) "unique_trades" could be summed over each day, over each week, over each month or over each year.

Because the statistics depend upon the date intervals over which they are computed, the field structure is naturally relational with the date interval associated to the list of statistics as an array in the JSON format. This is flattened in the CSV format.

Table 1 summarizes which parameters can be associated with which web service queries. N/A means that the parameter is not available for that query.

Table 1 Web Service Query Parameters

Parameters → Queries ↓	api_key	format	start-end date-times	aggregate_level	exclude_public
comment	required	required	optional	N/A	optional
person	required	required	optional	N/A	optional
person/leaderboard	required	required	optional	N/A	optional
question	required	required	optional	N/A	optional
question_history	required	required	optional	optional	optional
trade_history	required	required	optional	optional	optional

Query	Field	Definition
comment		
	user_id	User ID of the person who made the comment
	down_votes	Number of down votes on the comment
	up_votes	Number of up votes on the comment
	spam_score	An internal scoring mechanism to determine if the item is spam
	is_spam	A Boolean flag if the spam_score has exceeded a spam_score_threshold
	trade_id	The trade (if any) this comment is tied to
	comment_text	The html encoded text for the comment
	is_alert	A boolean flag indicating if this comment was flagged to alert administration
	created_at	The date-time that the comment was created in yyyy-mm-ddThh:mm:ss format
	parent_comment_id	A parent comment (if any) to which this comment is a reply
	comment_id	The internal id of the comment
	question_id	The question to which this comment refers
person		
	username	User name of the specified user
	created_at	When the user was created in YYYY-MM-DDThh:mm:ss format
	interests	Any interests (comma separated text strings) specified at registration
	is_active	True if account is currently active
	default_trade_preference	Specifies if user has indicated if they want the safe (1) or power (2) trade mode as their default, Null will use safe mode
	about_me	Freeform text provided by user for display
	opt_out_email	Flag indicating if they have opted out of any emails (null or 0 indicates all email is ok)
	num_trades	Total trades performed by user
	user_id	Internal user_id
	groups	List of groups to which this person is associated
	referral_id	Code string indicating if this user registered as part of a campaign or other referral code or group
	email_verified	If this user has verified/confirmed the email address used in the system
person/leaderboard		
	sampled_at	The date-time when the max_score was obtained in YYYY-MM-DDThh:mm:ss format
	user_id	Internal user_id
	max_score	The highest leaderboard score during the start and end date range
question		
	created_at	The specific date-time the question was created in yyyy-mm-ddThh:mm:ss format
	is_visible	True of null if question is visible
	resolution_index	Null if open, the index of the choice settled at if question is resolved (or highest value if mixture)
	resolution_value_array	Array by choice index of the resolution such that the sum equals 1. If this is a non mixture settlement one array element will be 1 (the resolved choice index) and other array elements will be 0
	resolution_at	Null if question is still open , else this is the specific date-time at which the question was resolved (yyyy-mm-ddThh:mm:ss format)

	pending_until	Null if question is open , else a specific time if the question should be considered pending resolution and no trading allowed after this time(yyyy-mm-ddThh:mm:ss format). This is essentially a locking mechanism, for example used when the resolution of a question is in dispute or review
	keywords	Freeform list of keywords set for question
	question_id	Internal id of the question. A unique id that can be used to refer to associate this question to others returned in any query, such as question_history or trade_history
	relationships	A list of link structures consisting of triples of the form (source_question_id, relationship_kind, destination_question_id) together representing a graph of questions
	source_question_id	The parent question in a link relationship
	relationship_kind	Bayesian network assumptive dependency relationship; other relationship types are TBD
	destination_question_id	The child question in a link relationship
	priority	A numeric ranking of this questions priority used for sorted display (display sorting goes from high to low)
	type	Question choices – currently only binary or multi-valued
	short_name	A symbol or short name to refer to the question
	question_contributors	A list of user_ids of persons who contributed to the development of this question
	last_traded_at	Null if never traded, otherwise the specific date-time the question was last traded (yyyy-mm-ddThh:mm:ss format)
	categories	A string separated list of category names to which this question is associated
	question_text	HTML freeform text description for the question (long)
	name	Textual description of the question
	groups	Private groups in which this question is included if applicable to the question
	is_locked	True if the question is locked for trading
	challenge	A question challenge area id if any is set for the question
	choices	If multiple choice, this is a list of the questions choice index names. If binary it is blank.
	spark_id	The corresponding internal identifier (if any) of the item in the spark system that corresponds to this question
	is_ordered	A flag indicating if the question multiple choice answers should be considered ordered for purposes of scoring
	spark_author	Who created the question including their spark_id, spark_username, predict_id, and predict_username – currently the raw json
	spark_collaborators	Who collaborated on the question including their spark_id, spark_username, predict_id, predict_username, number of edits, number of chats – currently the raw json
	spark_created_at	when the Spark question was created

	spark_published_at	when the Spark question was published to Predict
	spark_publisher	who published the question to Predict - including their spark_id, spark_username, predict_id, and predict_username
	raw_serialized_model	JSON encoded serialized model representing the scaled mapping information
	classification	What do we classify this question as for scoring and other purposes: binary, scaled, unordered multinomial , ordered multinomial , share
	marginal_prob_at_settlement	What was the probability at initial settlement
question_history		
	probabilities	Ordered list (by choice index) of the probability of each choice index (must sum to 1)
	sampled_at	The date-time this probability existed (yyyy-mm-ddThh:mm:ss format)
	question_id	A unique id that can be used to refer to associate this question to others returned in any query, such as question or trade_history
	aggregate_time_period	The date interval over which a statistic is computed - only returned when the aggregate_level parameter is used
	unique_questions	The number of questions on Predict during the time_period - only returned when the aggregate_level parameter is used
	mean_statistics	The average number of unique_questions on Predict during the time_period - only returned when the aggregate_level parameter is used
	stdev_statistics	The standard deviation of the number of unique_questions on Predict during the time_period - only returned when the aggregate_level parameter is used
trade_history		
	trade_id	Unique ID assigned to each trade
	updated_at	The date-time when the trade was last updated for any reason (including when the question resolves) in YYYY-MM-DDThh:mm:ss format
	old_value_list	A list of probabilities by choice index (must sum to 1.0) for the question and current assumptions prior to the trade
	new_value_list	A list of probabilities by choice index (must sum to 1.0) for the question and current assumptions after the trade
	assets_per_option	The resulting change in assets by choice index for each possible settlement
	recurring_trade_id	What recurring trade rule generated this trade (if any)
	generated_for_recurring_trade_id	Was this a template for a recurring trade rule (if so which)
	trade_status	The current status code for the current trade : Null if live, -1 if some condition has made this trade no longer relevant, resolution choice index if question is resolved (if mixture resolution will be highest resolution choice index)
	asset_resolution	After resolution the change in the users assets as a result of this trade
	user_id	The id of the user who performed this trade
	traded_at	The date-time when the trade was executed in YYYY-MM-

		DDThh:mm:ss format
	choice_index	The index of the choice the trade focused on (in multiple choice we trade only on a single choice index normally). If binary will be 1 for true choice index
	serialized_assumptions	The assumptions as a list of question/choice index, applied with this trade
	question_id	A unique id that can be used to refer to associate this question to others returned in any query, such as question or question_history
	interface_type	The interface type used to make the trade, 1 indicated safe mode, 2 indicates power mode. There are some administrative trades where this is blank
	raw_user_selection	A json string corresponding to the actual trade requested by a user when in safe mode or if this is a scaled question
	aggregate_time_period	The date interval over which a statistic is computed- only returned when the aggregate_level parameter is used
	unique_questions	The number of questions on Predict during the time_period - only returned when the aggregate_level parameter is used
	num_trades	The number of trades made on Predict during the time_period - only returned when the aggregate_level parameter is used
	num_traders	The number of traders on Predict during the time_period - only returned when the aggregate_level parameter is used
	mean_statistics	The average number of unique_questions, unique_trades or unique_traders on Predict during the time_period - only returned when the aggregate_level parameter is used
	stdev_statistics	The standard deviation of the number of unique_questions unique_trades or unique_traders on on Predict during the time_period - only returned when the aggregate_level parameter is used
	old_marginal	If this is an assumptive question what was the old marginal before the trade, only valid for new trades
	new_marginal	If this is an assumptive question what is the new marginal after the trade, only valid for new trades

4. Field Definitions

Generally speaking the JSON (http://www.json.org) file option is preferred because of hierarchical, nested or relational data structures, such as probability distributions which pair a probability value with each of a forecast question's possible choices. XML (http://www.w3.org/XML/) format is available as well. For those who prefer to work in spreadsheets or are unfamiliar with the JSON and XML file formats we do provide a flattened CSV (http://tools.ietf.org/html/rfc4180) download option.

Table 2 defines the fields for the records returned for each query. These parameters may evolve over time so check http://datamart.scicast.org/api_docs/ or contact support@scicast.org if you encounter any issues.

Relational structures are not shown in the table. Instead each atomic field element is defined. The structural relations can be observed by examining the downloaded file.

Table 2 Query Field Definitions

5. Access Methods

Warning: Individual fields and total file sizes can be arbitrarily large depending on the query executed. If you import a CSV file into a spreadsheet it is possible to overflow admissible size limits. If this proves to be an issue for you, we first recommend you consider restricting your date range parameters. If this does not solve your problem, please contact us at support@scicast.org.

Access can be performed, by browser, by command line or by writing your own script.

You can type or paste a web service query into a browser just like any URL. Web service queries with JSON option will return the file directly into the browser window. Web service queries with CSV option will pop up a window giving you the option to save or open the returned file, depending on your browser type and how it is configured to handle CSV file types.

A program like cURL (http://curl.haxx.se/) can be used in command line to execute a web service query.

A library of Data Mart access scripts, including the one below, is available at https://github.com/SciCast/datamart_scripts .If you develop scripts or codes with new capabilities or in different languages we urge you to add them to the GitHub repository for others to re-use. Thanks!

The following Python script takes the api_key as a parameter. The web service query itself, this one about question_history, is hard coded.

The filename for the script is "simple_question_query.py". The script would be invoked from a directory called "Python_Scripts" using the command:

Python_Scripts/simple_question_query.py <api_key>.

The following script is freely available for non-commercial use.

```
#!/usr/bin/env python
#
#  Copyright (c) 2013 Gold Brand Software, LLC.
#  This software was developed under U.S. Government contract number
D11PC20062.
#  The U.S. Federal Government is granted unlimited rights as defined in FAR
52.227-14.
#  All other rights reserved.
#
class DatamartRetrievalException(Exception):pass

import requests
import sys

def get_question_as_json(api_key):
    location =
"http://datamart.scicast.org/question/?format=json&api_key=%s"%(api_key)
    r = requests.get(location)
```

```
    if r.status_code != 200:
        raise DatamartRetrievalException()
    return r.json()

if __name__ == '__main__':
    if len(sys.argv) > 1:
        api_key = sys.argv[1]
        json_data = get_question_as_json(api_key = api_key)
        print "Total Questions: %s"%(len(json_data))
        for q in json_data:
            print "%s,%s\n"%(q['id'],q['name'])

    else:
        print "Usage: %s api_key"%(sys.argv[0])
```

6. Quick Start

The following instructions will give you web service queries without filtering parameters that will include all the data for the query type since the beginning of SciCast. For parameter options see section 2.2. For the (atomic) field definitions for the returned records see section 3.

1) Email support@scicast.org and request Data Mart Access.
2) You will receive a URL for a Data Mart access request web form in email; fill it out and push the "Submit" button.
3) When your request information is validated you will receive an api_key by email. Please do not share the api_key. If you don't get an api_key or other response within three days, please email an inquiry to support@scicast.org.
4) Type web service queries directly into your browser in the form:

   http://datamart.scicast.org/<query_type>/?format=<file_type>&api_key=<api_key>

5) The options are:
   a. **query_type**: comment, person, person/leaderboard, question, question_history, trade_history
   b. **file_type:** json (preferred), csv or xml

Engineering documentation is available at http://datamart.scicast.org/api_docs/

Various scripts, codes and documentation are available at https://github.com/SciCast/datamart_scripts.

For help email support@scicast.org.