Setting up Mechanical Turk, *boto(3)*, and *mturkutils*

Florian Jaeger with help from Andrew Watts, Linda Liu, Zach Burchill, and Xin Xie

November 30, 2020

Contents

1 Set up Mechanical Turk accounts

If you are not already familiar with Amazon Mechanical Turk, the steps below will help you get started:

1. Register for an MTurk requester account (https://www.mturk.com/mturk/welcome),
   sandbox requester account (https://requestersandbox.mturk.com),
   sandbox worker account (https://workersandbox.mturk.com),
   and an Amazon Web Services (AWS) account (https://aws.amazon.com/).
   - You can use the same email address that you use for your main Amazon.com account to make things easier, but you might want to make a separate account (if you plan to share your password with other members of the group).
   - For the AWS account, you should select Basic access when prompted. Note that you will be asked to enter a credit card (though you will not be charged for this class).
   - **Link your AWS account with your MTurk requester account**. To do that, go to `https://requester.mturk.com/developer`. Your AWS account should already show under Step 1 of that page. Under Step 2 of that page, there is an orange button "Link your AWS account" to link the requester account to AWS. Do that.
   - **It may take up to a few days before your account(s) are confirmed, so be sure to sign up early.**
   - **If you have a non-U.S. address, your account may be denied.**
2. Sign in to your AWS account and navigate to the console (https://console.aws.amazon.com/iam/home). There will be a blue button on the upper left that says "Create New Users". It doesn't matter what you enter for your username, so pick whatever you want. Once you create a user, you will be provided

with an access key and a secret key. **These are important; please save them somewhere and do not share the secret key with others**!

3. Now you need to give your user permissions to use MTurk. On the left hand side of the console dashboard, you should see a tab called Policies. Click on that and select the one that says AmazonMechanicalTurkFullAccess. On the top of the page, select Policy Actions, and select Attach. You should be shown a page that contains the username from the step about. Select that username to give it MTurk permissions.

2 Install Python and boto3

1. Begin by install Python (version 3 or higher). **If you had an earlier version of Python installed, the new version might be linked as Python3 and pip might be linked as pip3, and you might have to adjust the commands below accordingly.** pip helps you install Python packages. It should come with any recent version of Python. You can find out what version of Python you have by typing the following into a terminal window:

   ```
   python --version
   ```

   or, if you already have python version 3, try:

   ```
   python3 --version
   ```

2. Then install boto3, a Python package for interaction with Amazon's MTurk.

   ```
   pip install boto3
   ```

3. You will also need to install a few Python packages. pyyaml is a YAML parser for Python. YAML stands for "Yet Another Markup Language" or "YAML Ain't Markup Language", and is a convenient way of reading (for humans and computers) data. You will see an example of YAML syntax in a later section. In your terminal:

   ```
   pip install pyyaml
   pip install ruamel_yaml
   pip install xmltodict
   pip install unicodecsv
   pip install pandas
   ```

4. And if you're using Python 3.9 or higher additionally install the following:

   ```
   pip install ruamel.yaml.clib
   ```

2.1 If you want to use boto3 through R

Finally, install pyMTurkR, an R library that allow interaction with MTurk through the boto package without requiring you to know Python. Step by step instructions are provided at https://cran.r-project.org/web/packages/pyMTurkR/readme/README.html. Make sure to complete all three steps on that page.

3 Link Boto to your AWS account

Now we need to link boto with your account by setting a configuration file with your AWS access key and secret key from Step 1. This can be done in a number of ways. The ultimate goal in all cases is to set three environment variables AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY, and AWS_DEFAULT_REGION (to: us-east-1).

On Windows, directly set all three variables by editing your system environment variables. On Mac or Linux machines, create a credentials file in ~/.aws (i.e. ~/.aws/credentials) and edit it to contain the following lines (case matters!):

```
[default]
aws_access_key_id = your_access_key_here
aws_secret_access_key = your_secret_key_here
```

Next, create and edit the config file in ~/.aws (i.e. ~/.aws/config):

```
[default]
region = us-east-1
```

Both the shared credentials file and the config file also supports the concept of profiles. Profiles represent logical groups of configuration. For example, for the shared credential file you might want to specify two separate profiles:

```
[default]
aws_access_key_id=foo
aws_secret_access_key=bar

[lab]
aws_access_key_id=foo2
aws_secret_access_key=bar2
```

You can then directly refer to the name of a profile when you work with *boto3* or *mturkutils*. This is convenient, for example, if you want to use your personal AWS account for debugging your experiment but then use a user linked to the lab's AWS account for the actual experiment. For more information or if you run into problems, see https://boto3.amazonaws.com/v1/documentation/api/latest/guide/credentials.html.

If you're stuck on how to create the config file...

You can use your text editor of choice to create this file. If you're stuck or don't have a text editor preference, here are some instructions for how to do it using vi. This is the default text editor for all UNIX-esque systems, so if you have a Mac, you already have it. As a warning, if you haven't used vi before, these steps might seem funky and unintuitive. But vi is pretty much as fast, lightweight, and powerful as text editors come, so I highly recommend it.

At the command line, type

```
    vi ~/.aws/credentials
```

This opens vi with a new file called ~/.boto. Now that you are in vi, do the following steps:

- Press i (to go into insert mode)
- Paste your credentials in the proper format
- Press Esc (to go into command mode)
- Type :wq Enter (to quit and save)

3.1 AWS shell

AWS shell is a tool that simplifies interaction with MTurk through the command line. There is a nice introduction to AWS shell and a tutorial on more advanced topics.

3.2 If want to use boto3 through R

Open the .Renviron file. You can do this through RStudio, either through the file menu (make sure to selection the option ”Show hidden files” or through the terminal within RStudio). On Linux/MacOS, the file will be in your home directory. On Windows, the .Renviron file might be under C:\Users\USERNAME\Documents\. If you do not already have a .Renviron file, you can make one in RStudio (File > New File > Textfile and save it as .Renviron in the directory indicated in the previous sentences). In the .Renviron file add the following lines and save the file (case matters!):

```
AWS_ACCESS_KEY_ID=your_access_key_here
AWS_SECRET_ACCESS_KEY=your_secret_key_here
```

Then restart R (from the Session menu in RStudio). Your AWS credentials should not be linked to boto3 everytime you use R.

4 Test it

Let’s make sure everything is properly connected. To do this, we will try to connect to your MTurk sandbox using boto3. The sandbox is the testing area for HITs; only requesters (and not normal workers) can view sandboxed HITs. Other than that, the sandbox environment is identical to the environment that workers see. We will explore the sandbox in more detail in Section 6 by uploading a HIT to it.

In your command line, start python (or use whatever you typically use to run python). In your Python interpreter, type the following to connect to MTurk sandbox:

```
import boto3
session = boto3.Session()
# Alternatively, specify which profile you would like to use, e.g.:
# session = boto3.Session(profile_name='lab')

mtc = session.client('mturk')

# Alternatively, you can specify a specific region name, overriding your default profile
# endpoint = 'https://mturk-requester-sandbox.us-east-1.amazonaws.com'
# mtc = session.client('mturk', endpoint_url=endpoint, region_name='us-east-1')
print(mtc.get_account_balance())
```

You should get something like the following output (MTurk sandbox always contains $10k of fake money):

```
{'AvailableBalance': '10000.00',
'ResponseMetadata': {
    'RequestId': 'XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX',
    'HTTPStatusCode': 200,
    'HTTPHeaders':
        {'x-amzn-requestid': 'XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX',
        'content-type': 'application/x-amz-json-1.1',
        'content-length': '31',
        'date': 'Tue, 13 Oct 2020 19:02:08 GMT'},
    'RetryAttempts': 0}}
```

Be sure to get this working before continuing!

4.1 If you're using boto3 through R

In RStudio:

```
library(pyMTurkR)
getbalance()
```

You should get the following output (MTurk sandbox always contains $10k of fake money):

```
[$10,000.00]
```

If you instead get an error message like the following:

```
Error in py_call_impl(callable, dots$args, dots$keywords):
RequestError: An error occurred (RequestError) when calling the GetAccountBalance operation:
To use the MTurk API, you will need an Amazon Web Services (AWS) Account.
Your AWS account must be linked to your Amazon Mechanical Turk Account.
Visit https://requestersandbox.mturk.com/developer to get started.
     Check your AWS credentials.
 Error in py_call_impl(callable, dots$args, dots$keywords) :
  RequestError: An error occurred (RequestError) when calling the GetAccountBalance operation:
  To use the MTurk API, you will need an Amazon Web Services (AWS) Account.
  Your AWS account must be linked to your Amazon Mechanical Turk Account.
  Visit https://requestersandbox.mturk.com/developer to get started.
```

This points to some issue with your account setup. PyMTurkR is able to reach your Amazon's IAM (Identity and Access Management) system with your credentials, but either a) your credentials are wrong, or b) you forgot to create a requester sandbox account under Step 1, and/or c) you forgot to link the MTurk requester account to AWS under Step 1.

5 Interacting with MTurk via mturkutils

There are many ways to interact with MTurk. One convenient way is to use a collection of boto3 script written by former HLP lab manager Andrew Watts. You can get the most recent mturkutils at https://bitbucket.org/hlplab/mturkutils/src/master/. Alternatively, you might use the R library pyMTurkR.

This tutorial comes pre-packaged with mturkutils (version as of October, 2020), and we will use it below to upload HIT to MTurk and obtain results from MTurk. The mturkutils scripts can be found in the mturkutils/ folder. **In your terminal, navigate to the mturkutils directory.**

As you can read in the README.md "The original scripts use the now unsupported boto library. Some scripts have been updated to use the current boto3 and botocore libraries. Originals are in the ‘boto‘ directory and updated are in ‘boto3‘. Updated versions also require Python 3." We will be using the scripts in the boto3 folder.

To learn more about any of the mturkutils scripts, call the script with the -h tag ("h" for help). In your terminal, go to the mturkutils/boto3/ folder and try, for example:

```
python loadHIT.boto3.py -h
```

Make sure that you are in the mturkutils/boto3/ folder when you run this (or provide the right path). Notice that loadHits.py takes a configuration file (our YAML file), and an optional sandbox tag. **Unless you're ready for your experiment to go live and pay for it, always make sure to use the sandbox tag! Once a hit is posted to MTurk via boto, it is exceedingly difficult to delete it. And, in any case, you will charged for it.**